



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

TESIS PROFESIONAL

BÚSQUEDA ASISTIDA POR COMPUTADORA

PRESENTADA POR:
KARLA VANESSA SALDAÑA CABRERA

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

DIRIGIDA POR:
M. en I. ALBERTO FUENTES MAYA



MEXICO, D.F FEBRERO DE 2008

Agradecimientos:

A mi mamá Laura y papá Carlos; por que sin ellos yo no podría estar aquí, agradezco su apoyo, regaños, consejos y amor, que dios los bendiga. Los quiero mucho.

A mi hermana Violeta; por ese carácter fuerte que tiene, se ha ganado toda mi admiración; la quiero.

No podría faltar agradecer toda mi familia, los que están aquí y los que me dejaron, que dios los cuide y los proteja.

A una persona muy especial; que me enseñó muchas cosas, me dijo algo que nunca olvidare: Tú puedes, confío en ti y se que eres capaz de hacer lo que quieras. Gracias. QUIRO.

A mis amigos del alma con quienes viví muchas cosas, y agradezco a la vida por ponerlos en mi camino. Elizabeth, Mónica, Elisa, y Rafael

No puede faltar mi profe; me dio una tesis que me gustó mucho, me ha dejado impresionada con lo que sabe y por los consejos que me dio. Gracias. Alberto Fuentes Maya. Lo admiro.

Y a toda esa banda que he conocido, que he aprendido y adquirido en mí su personalidad, es un gusto saber que todos somos diferentes, es el chiste de vivir.

A la Facultad de Ingeniería de la UNAM y a todos mis profesores, agradezco lo enseñado, los desvelos y el gran valor de sentirme universitaria.

ÍNDICE

Objetivo.....	i
Introducción.....	ii
Capítulo I “Introducción al Análisis de Operaciones Navales”	
1.0 Análisis de Operaciones.....	1
1.0.1 Métodos del Análisis de Operaciones.....	1
1.1 Análisis de Operaciones Navales.....	2
1.2 Método del Análisis de Operaciones Navales.....	2
1.2.1 Planteamiento del Problema.....	3
1.2.2 Solución del Problema.....	6
1.2.3 Comunicación de Resultados.....	7
Capítulo II “Teoría de búsqueda y detección“	
2.0 Teoría de Búsqueda	8
2.1 Teoría de Detección	9
2.1.1 Generalidades Concernientes a la Detección.....	9
2.1.2 Hechos Básicos que se Consideran en todo Tipo de Búsqueda.....	10
2.2 Clasificación General de Tipos de Búsqueda	11
Capítulo III “La Decisión”	
3.0 Grados de riesgo.....	16
3.1 Criterios bajo incertidumbre.....	17
3.2 Métodos de Probabilidad de Búsqueda y Detección.....	17
Capítulo IV “Aplicación de búsqueda y detección en superficie”	
4.0 Problemas de Aplicación y Simulación.....	25
4.0.1 Búsqueda Rectangular.....	25

4.0.2 Expansión Cuadrada.....	33
4.0.3 Sector Base Fija.....	36
4.0.4 Sector Base Móvil.....	42
4.0.5 Búsqueda por Delante.....	50
4.0.6 Búsqueda por Detrás.....	57
4.0.7 Búsqueda por Flanco.....	65
Capítulo V “Búsqueda Asistida por Computadora”	
5.1 Generación de Números Aleatorios.....	73
5.1.1 Método Monte Carlo.....	73
5.2 Implementación.....	74
5.2.1 Simulación del CAS (Búsqueda Asistida por Computadora).....	80
Conclusiones.....	85
Anexo A.....	86
Anexo B.....	89
Bibliografía.....	138

OBJETIVO

Analizar y aplicar el Análisis de Operaciones Navales, en problemas de búsqueda y detección en superficie; con la finalidad de desarrollar una herramienta de cómputo que auxilie al tomador de decisiones para la solución de problemas de búsqueda y detección en superficie.

INTRODUCCIÓN

El análisis de operaciones (AO), se inició como respuesta a necesidades militares en la segunda guerra mundial, y se le llamaba *análisis o evaluación de operaciones*. Aunque desde hace tiempo estas técnicas y métodos de aproximación se usan en la industria, gobierno y actividades militares, es en esta última en donde su aplicación es predominante. Sin embargo, las técnicas del AO ayudan a la toma de decisiones en cualquier campo de desarrollo, sea este industrial, gubernamental o militar.

El AO es un método científico, con una metodología definida para atacar nuevos problemas y encontrar soluciones específicas.

El personal que emplea el análisis de operaciones como herramienta de solución para resolver problemas, en su campo de trabajo, tiene la tarea de presentar los aspectos cuantitativos en una forma comprensible y resaltar, si es posible, algunos de los aspectos no cuantitativos que pueden requerir la consideración de personal importante antes de que tome una decisión.

Por eso, el análisis de operaciones debe proveer al personal directivo de bases cuantitativas para la posible decisión en donde éste se de cuenta de la gran importancia que tienen las técnicas en la presentación clara de los resultados. De hecho, la mayor contribución del investigador de operaciones es decidir cuál es el problema real. Y tomar en cuenta que todo método científico implica compartir sus resultados con otros; sin olvidar que el directivo es el que tomará una decisión.

En numerosos problemas, la experiencia del directivo y su juicio, son suficientes para predecir con cierta seguridad los resultados de cada alternativa sopesando sus méritos relativos.

En el ámbito naval, las operaciones navales han llegado a ser tan complejas como el propio equipo utilizado. En la actual generación, el proceso de toma de decisiones, en específico en búsqueda y detección de una embarcación, ha pasado a ser, no solamente una de las funciones más importantes de cualquier operación naval, sino que ha alcanzado la importancia de una ciencia con la cual se debe estar familiarizado.

Es importante mencionar que existen problemas de búsqueda y detección terrestre, marítima y aérea, siendo el interés principal del presente trabajo, únicamente los problemas marítimos (superficie), debido a la importancia y recursos con los que cuenta México, en la actualidad.

La teoría de Búsqueda y Detección, es un recurso relevante en el Análisis de Operaciones Navales que se implementa para el proceso de Toma de Decisiones. El principal propósito del presente trabajo, es desarrollar una herramienta computacional para la toma de decisiones en problemas de búsqueda y detección.

La estructura general de la tesis consiste en cinco capítulos, y una sección en la que se muestra la conclusión. En el Capítulo I; se da una introducción al AO y sus métodos; así como una inducción al Análisis de Operaciones Navales con sus técnicas más utilizadas. En el Capítulo II; se explican los conceptos básicos de la teoría de Búsqueda y Detección, enfatizando en los tipos de búsqueda existentes y su clasificación. En el Capítulo III; se describe la importancia del AO en la toma de decisiones, y la relación de complicidad que debe haber entre el analista y el directivo que toma la decisión, basándose en aspectos medulares. El Capítulo IV; se muestra la simulación de los distintos tipos de búsqueda y detección, basándose en problemas reales de aplicación. El Capítulo V; se expone la implementación de un Sistema de Búsqueda Asistida por Computadora, que utiliza el método Monte Carlo para la generación de números aleatorios, los cuales nos ayudarán a hacer las iteraciones de escenarios, mediante métodos probabilísticos, dando como resultado la inferencia en tiempo y posición del objeto buscado. Y finalmente se presentan, las conclusiones del proyecto, describiendo su utilidad y visión en un futuro.

Capítulo I

Introducción al Análisis de Operaciones Navales

El análisis de operaciones es un método científico que proporciona a los departamentos ejecutivos, bases cuantitativas para la toma de decisiones respecto a las operaciones bajo su control.

Al tomar una decisión debemos basarnos no sólo en la experiencia en sí misma, sino en cómo adaptarla con el fin de obtener resultados óptimos. Para resolver problemas se requiere, además de una mente clara y lógica, un planteamiento del problema que nos ayude a obtener las conclusiones correctas, además de evitarnos el olvido de cualquiera de los elementos principales que intervienen en tal problema. Han existido a lo largo del tiempo numerosos métodos que ayudan al planteamiento y obtención de la solución de un problema determinado. Estos métodos incluyen una evaluación de la situación numerando todos los factores y su importancia relativa. Posteriormente, se estudian las diferentes alternativas para conseguir el objetivo, llegando finalmente a la elección de la decisión adecuada.

1.0 Análisis de Operaciones

El Análisis de Operaciones o Investigación de Operaciones, como comúnmente se le conoce en el ámbito civil, es un método científico, con una metodología definida, que puede ser enseñado y transmitido para atacar nuevos problemas y encontrar soluciones específicas.

1.0.1 Métodos del Análisis de Operaciones

La metodología del Análisis de operaciones, se relaciona con el tipo de datos que pueden ser obtenidos mediante estudios; por lo general resulta que pocos datos numéricos son comprobables en fenómenos de gran complejidad. Los problemas son por lo tanto, más cercanos a problemas biológicos y económicos que a problemas físicos, donde usualmente los datos numéricos son de fenómenos relativamente simples y comprobables. Sin embargo, la investigación de operaciones, así como cualquier otra ciencia no debe copiar a detalle los métodos técnicos de otras ciencias, sino que debe desarrollar sus propios métodos, a la medida de sus problemas y materiales. Entre estos métodos podemos mencionar los siguientes:

- a) Métodos Estadísticos.
- b) Tareas de campo, recolección de datos.
- c) Experimentos Operacionales
- d) Métodos Analíticos.

El ser un analista de operaciones no demanda ser un especialista en ninguna rama de la ciencia en particular. Sin embargo, necesita ser una persona con considerable experiencia

en *investigación* de naturaleza científica, ya sea física, biológica, matemática, o bien de cualquier otra ciencia.

1.1 Análisis de Operaciones Navales

En los años que siguieron inmediatamente a la guerra se hizo evidente que este trabajo no debía ser exclusivamente confiado a civiles. Tan capaz, técnicamente, como un científico, el oficial de marina posee una característica vital en la apreciación de las sutilezas de muchos problemas militares, así como, un espíritu más realista. Esto se debe a la mejor comprensión del conjunto del ambiente operacional, así como, su mejor apreciación de los factores gracias a su propia experiencia.

Cada vez se hizo más ostensible la diferencia entre el oficial naval que conocía el problema, pero no tenía un conocimiento científico suficiente para atacarlo analíticamente, y/o exponerlo en términos cuantitativos, y el científico civil, que tenía competencia matemática, pero no la necesaria profundidad en la apreciación de los factores sutiles del problema. Esta diferencia se hizo más evidente al poseer los oficiales de marina una gran experiencia personal, y se resolvió especializándose ellos mismos en el Análisis de Operaciones, o bien guiar a los Analistas Civiles en determinadas áreas, en las cuales, era necesario un excesivo conocimiento científico.

El número de científicos integrados en la organización de Análisis de Operaciones ha crecido, a la vez que, se han ido ampliando sus aplicaciones a un mayor número de operaciones navales. Hoy en día se tiene la certeza de que es esencial en la educación de todo oficial de marina el estudio de los fundamentos del Análisis de Operaciones, ya que, se puede decir sin temor a error, que su aplicación en los problemas navales irá en aumento con los años debido a que los sistemas de armas son cada vez más complejos, el costo de errores continuará creciendo exponencialmente y, finalmente, habrá que ir adoptando nuevos conceptos y técnicas que no han sido probadas en combate y que están siendo suministrados por los grandes avances tecnológicos.

1.2 Método del Análisis de Operaciones Navales

El Análisis de Operaciones consiste esencialmente en la aplicación de un método científico para resolver problemas operacionales. Sin entrar en discusión sobre lo que constituye un método científico, es razonable establecer que, allí donde se ha aplicado con éxito el Análisis de Operaciones, ha mostrado ser un modelo consistente.

A este modelo se le llama Método de Análisis de Operaciones. Esto no significa, de ninguna manera, que dicho método sea un algoritmo matemático “resuélvelo-todo” o una panacea, sino más bien, que se trata de un método que se halla en diferente estado de desarrollo según el problema que trate de resolver. El Análisis de Operaciones puede describirse en forma general con los siguientes términos:

- a) Planteamiento del problema
 - 1) Determinación de los objetivos de la operación
 - 2) Enumeración de todas las alternativas
 - 3) Medición de la efectividad para cada una de estas alternativas
 - 4) Determinación y numeración de todas las variables que deben considerarse.
- b) Solución del problema: uno solo o combinación de los siguientes puntos:
 - 1) Análisis teórico
 - 2) Análisis estadístico de los datos empíricos
 - 3) Juicios o ejercicios controlados
 - 4) Simulación, investigación experimental. Utilizando modelos físicos, analógicos y digitales, de la situación operativa.
- c) Comunicación de los resultados obtenidos.

1.2.1 Planteamiento del Problema

“El planteamiento de un problema es reducirlo a sus aspectos y relaciones fundamentales a fin de poder realizar su estudio intensivo”¹

- a) ¿Cuál es el objetivo a alcanzar?
- b) ¿Cuáles son las diferentes alternativas disponibles?
- c) ¿Qué factores (variables) contribuirán al éxito o fracaso de cada alternativa?
- d) ¿Qué unidad se puede y debe usar para medir y comparar la efectividad de cada alternativa?

El planteamiento de un problema consiste principalmente, en contestar a estas preguntas básicas, que no son tan triviales como parecen a primera vista. Las respuestas, frecuentemente, parecen no ser expresables, o, cuando se expresan, son motivo de amplias discusiones; y en caso de problemas mayores no merecen solo un estudio superficial sino una profunda investigación, después del cual, para obtener la solución del problema pueden ser solamente necesarios unos pocos cálculos, o bien, coleccionar un número suficiente de datos.

El objetivo de la operación real, es a menudo, una de las tareas más difíciles de estos problemas. El objetivo que debe tenerse en cuenta, es de la persona que tiene la responsabilidad de la decisión; el objetivo de un tomador de decisiones puede determinarse, a menudo, estudiando la misión que le ha dado su superior.

En los problemas que involucren una decisión de alto nivel estratégico, el análisis puede ser dirigido por los consejeros del ejecutivo o por una empresa de investigación

¹ Jose Luis Lopez Cano;

independiente. En ambos casos se debe determinar con toda precisión cuál es el objetivo. Al principio este objetivo puede ser vago pero se puede identificar con toda claridad.

A veces se confunde el objetivo con la acción que debe realizarse para cumplirlo, la ayuda del analista debe consistir en proporcionar al directivo la base segura, y no tendenciosa, que sirva para tomar una decisión.

Las diferentes alternativas; mientras que el objetivo debe de ser determinado siempre en primer lugar, los demás pasos en el planteamiento del problema no siguen siempre una secuencia definida. Aún más, el analista debe considerar cada paso en función de los demás, yendo de uno a otro y de regreso al primero. La primera pregunta que se plantea cuando se quiere conseguir una lista de alternativas es ¿Qué limitaciones tiene mi observación? ¿Puedo considerar alternativas que involucren un incremento en hombres, material o equipo, o me estoy limitando a diferentes tácticas y métodos de operación utilizando exclusivamente los recursos que tengo a la mano?

A veces, es fácil contestar después del estudio de las variables que intervienen en el problema, pero con frecuencia surge un problema inmediato teniendo un límite en las alternativas que se pueden utilizar rápidamente y el tiempo que se dispone para su análisis. En muchas organizaciones existen reglas sociales y morales que eliminan aquellas alternativas que suponen prácticas deshonestas e ilegales o bien que violan las costumbres y normas de vida.

Un procedimiento de estudio, sería hacer una lista de todas las opciones, con las cuales puede esperarse razonablemente alcanzar el objetivo. Este procedimiento disminuye la posibilidad de olvidar cualquier alternativa. En algunas ocasiones, un individuo que está estrechamente relacionado con la operación, encuentra grandes dificultades en la búsqueda de nuevas alternativas diferentes a aquellas que ha usado previamente. Esto puede resolverse, si en vez de uno hay varios individuos con experiencia y conocimientos adecuados del problema a considerar. No existe una forma establecida para descubrir buenas alternativas, aunque las condiciones particulares de un individuo, su ingeniosidad y experiencia, contribuirán probablemente en gran medida. Obviamente, se presenta una gran dificultad cuando el número de alternativas es tan grande que, hacer una lista completa llega a ser tedioso o imposible. Cuando nos enfrentamos con este problema es conveniente distribuir las alternativas en grupos.

Para comprobar que es válida una alternativa, basta considerar, si ella misma contiene un significado completo de manera que se alcance el objetivo. Cualquier alternativa que se plantee, solamente parte de los medios necesarios para alcanzar el objetivo, y no se considera ni completa ni válida.

Las variables; si realizamos una lista con todas las variables o factores que intervienen en un problema determinado, la podemos utilizar para cumplir determinados propósitos:

- a) Sirve como una guía para determinar los datos que deben obtenerse antes de resolver un problema.
- b) Indica la complejidad del problema y nos proporciona una ayuda para determinar el método o métodos, que se utilizarán en su análisis.

- c) Evita que un elemento importante del problema no sea considerado en su estudio.
- d) Proporciona una guía que nos ayuda a no omitir ninguna alternativa.
- e) Incluye todas aquellas cantidades que son útiles para obtener la “Medida de Efectividad” MOE (Measure Of Effectiveness)²

Las variables pueden ser clasificadas en diversas categorías. Unas pueden controlarse o al menos, el tomador de decisiones puede tener algún grado de influencia en ellas. Sobre otras variables puede que no tengamos ningún control. De éstas, algunas tienen valores conocidos y otras pueden estimarse. En general, los resultados de una operación dependen o son función de muchas variables. Simbólicamente se puede expresar como:

$$\text{Resultados} = f (X_1 , X_2 , X_3 , \dots X_n)$$

Tiene una importancia fundamental la definición de todas las variables que intervienen en el problema a considerar. Algunas de ellas podrán ser eliminadas a posteriori, si se demuestra que afectan insignificamente a los resultados. Por medio de este procedimiento podremos ir reduciendo el estudio a lo estrictamente esencial.

Al resolver el problema debemos realizar un “análisis de sensibilidad”³, con el fin de determinar aquellas variables de las cuales depende principalmente el resultado. Debe tenerse especial cuidado de que los valores asignados a estas variables, fundamentalmente estén determinados o estimados con la mayor precisión posible.

Las variables que se han discutido y considerado hasta este momento, son aquellas cuyos valores pueden ser cuantificados. Existen naturalmente otras variables cuya propia naturaleza las hace muy difíciles (sino imposibles) de cuantificar, como por ejemplo; la moral, el entrenamiento, el clima político, la salud, la inteligencia, la educación, etc.

La Medida de Efectividad (MOE, Measure Of Effectiveness). La base de la decisión, esencialmente consiste en predecir y describir los resultados esperados de cada alternativa posible. La decisión del tomador de decisiones al elegir una alternativa, debe basarse en la comparación de los resultados estimados. Si éstos son descritos en términos cualitativos, no siempre será posible considerar cada alternativa en función de su efectividad. Surgiría siempre la pregunta: “¿En qué medida es mejor una alternativa que la otra?”. Esta pregunta implica la necesidad de utilizar un método cuantitativo que nos exprese los resultados en cifras. Muy pocas veces es fácil predecir los resultados de una operación y aún más, expresar estos en una forma cuantitativa. La medida utilizada para comparar la efectividad de cada alternativa en función del objetivo se llama “Medida de Efectividad” (MOE).

² Medida de Efectividad: Es una medida utilizada para comparar la efectividad (o sea la capacidad para producir el efecto deseado) de algún proceso, en este caso el de las variables y alternativas de un problema.

³ Análisis de sensibilidad (que pasa si): Este tipo de análisis muestra lo que sucede si se hace determinado cambio en una de las variables de decisión. El objetivo es ver la forma como se afecta la medida de desempeño; es decir que tan sensible es la utilidad ante el cambio.

Por ejemplo; en un problema cuyo objetivo sea detectar un objetivo, las posibles medidas de efectividad serían la “probabilidad de detección” o el “tiempo esperado de detección”.

No hay que olvidar que, la MOE debe estar estrechamente relacionada con el objetivo de la operación. El número de submarinos hundidos por mes puede ser una MOE satisfactoria si el objetivo es destruir submarinos, sin embargo, si el objetivo fuese proteger a las unidades de superficie, la mejor alternativa podría ser una en la que quizás se hundiesen menos submarinos.

Otra de las características necesarias de la MOE es que debe ser medible. Debe existir siempre la posibilidad de que, teniendo suficientes datos y haciendo los cálculos necesarios, se pueda determinar su valor para toda alternativa. Aún más, ya que el objetivo es proporcionar una base útil para la decisión, estos cálculos deben poder realizarse antes de tomar la decisión, o, al menos, antes de que se ejecute la operación. Estos valores de la MOE son necesariamente valores de predicción. Una vez que se haya realizado la operación es posible comprobar su efectividad, pero desde luego, esta información a posteriori nos puede ayudar en la decisión original, pero tiene un gran valor para futuras decisiones.

En resumen la MOE debe ser:

- a) Medible
- b) Cuantificable y
- c) Claras en qué grado (real) se alcanza el objetivo deseado.

1.2.2 Solución del Problema

En el Análisis de Operaciones el término “solución” no significa lo que el término normalmente implica. No tiene por qué encontrarse una respuesta al problema, pues del análisis sigue una decisión automática. La solución consiste únicamente en medir la eficacia y determinar, por ejemplo, la predicción de su valor para cada alternativa.

Métodos de Solución. La solución involucra un análisis con el fin de predecir la eficacia de cada alternativa; y nos proporciona un valor cuantitativo de esta eficiencia. Los métodos utilizados en estas predicciones pueden clasificarse en cuatro categorías:

- a) Análisis teórico
- b) Análisis estadístico con datos empíricos
- c) Pruebas controladas o ejercicios
- d) Simulación

Normalmente el método de solución será una combinación de dos o más de estos métodos simples. Aún cuando un análisis sea esencialmente teórico, el analista se ve obligado a utilizar parámetros, cuyos factores son función de unos datos empíricos. El método que se utilice en un problema concreto, depende de muchos factores, incluyendo

entre ellos complejidad, cantidad de datos disponibles y la experiencia en problemas similares, sin olvidar el tiempo ni el talento para su solución.

Todos los métodos se apoyan fundamentalmente en la utilización de disciplinas matemáticas, tales como: álgebra, trigonometría, geometría, cálculo integral y diferencial, además del cálculo de probabilidades. Otros campos matemáticos, algunos de los cuales han sido desarrollados recientemente, encuentran asimismo aplicación en algunos problemas específicos. Estos campos son la estadística, programación lineal, confiabilidad, teoría de grafos, teoría de colas, econometría, cibernética, teoría de inventarios, teoría de decisiones, teoría de redes, programación dinámica, etc. Aunque las matemáticas proporcionan las herramientas básicas, no son, en sí mismas suficientes, ya que es necesario el conocimiento de los fenómenos físicos que intervienen en el problema, como por ejemplo, la propagación de las ondas sonoras en el agua, cuando se estudia un problema de detección submarina. **Por lo tanto, el análisis de un problema importante puede necesitar de matemáticos, físicos, economistas, psicólogos, ingenieros, etc.**

En general, la solución de un problema incluye el desarrollo de un modelo matemático que representa o simula la operación física. Esta técnica, no es menos peculiar que otras en el Análisis de Operaciones.

1.2.3 Comunicación de los Resultados

La solución de un problema no será completa, hasta que el tomador de decisiones haya sido comunicado de los resultados del análisis y cuente con más de una alternativa, de tal forma que le ayude a tomar una decisión. Muchas horas de tedioso análisis y el uso de las más brillantes técnicas matemáticas, pueden ser completamente inútiles si los resultados obtenidos no son entendibles para el tomador de decisiones, sino únicamente, por los propios analistas. Los informes de un equipo de análisis, son normalmente leídos y comunicados a un gran número de individuos o grupos, con el fin de que los resultados obtenidos puedan ser aplicados en diferentes campos. Algunos de estos grupos pueden necesitar una gran cantidad de detalles técnicos, mientras que otros, requerirán solamente una descripción funcional. Esto nos obliga a mantener un compromiso entre ambas necesidades; la solución yace en la propia organización del material con el que se realiza el informe en el uso discreto del lenguaje técnico.

Capítulo II

Teoría de Búsqueda y Detección

El objetivo de la búsqueda y detección es la localización de embarcaciones a la deriva en alta mar y la detección de barcos no nacionales que incursionan en aguas de patrimonio nacional.

2.0 Teoría de Búsqueda

La **búsqueda** involucra encontrar las balsas, botes salvavidas a la deriva, pequeños botes estancados en el agua y algunas veces personas a la deriva con chalecos salvavidas.

Se han desarrollado técnicas de predicción que pueden dar una posición bastante precisa del objeto de búsqueda; he aquí los elementos de la predicción.

- a) La velocidad del viento y de la corriente marina.
- b) Forma y tamaño del objeto perdido
- c) Estado de la mar.

Con esta información de deriva, se hacen predicciones de la posición del objetivo de búsqueda para intervalos de tiempo específicos.

El éxito de una búsqueda depende, o es afectado, por uno o varios de los siguientes factores:

- a) Naturaleza del objetivo (el objetivo puede ser cualquier tipo de unidad).
- b) Condiciones ambientales.
- c) Equipo de detección que se dispone.
- d) Táctica empleada.

Las condiciones ambientales consideradas en este tipo de operaciones navales, se representan no sólo por las condiciones oceanográficas, estado de la mar; condiciones de agua referidas a presión, temperatura, salinidad, etc.; fondo rocoso o aplacerado, sino también, por las condiciones climatológicas, iluminación dentro y fuera del agua, formación de nubes y muchos otros. Para superar cualquier cambio en las condiciones ambientales existentes y del objetivo, se agrupan los objetivos en los siguientes tipos básicos:

- a) Grandes
- b) Medianos y
- c) Pequeños.

Y las condiciones ambientales en:

- a) Mar Gruesa,
- b) Marejada y
- c) Calma.

Las características de los equipos disponibles juegan un importante papel en la búsqueda y detección. Cuanto más poderoso y sensible sea el equipo, mayores serán las probabilidades de llevar a cabo tales misiones. Cuanto mayor sea la confiabilidad y efectividad de los equipos, mayores serán las probabilidades de reducir las consecuencias de una acción.

2.1 Teoría de Detección

En las Operaciones Navales, la detección de buques, embarcaciones u objetos, en la actualidad ha pasado a ser, no solamente una de las funciones más importantes, sino que ha alcanzado la altura de una ciencia, con la cual debe estar familiarizado todo el científico dedicado a la detección de objetos de importancia en el campo donde se especialice. Cualquier acción debe ir precedida del conocimiento de su presencia y posición. Todo problema de detección depende de tres grandes factores:

- a) **Las características físicas** del instrumento de detección y del objetivo, junto con las condiciones físicas de propagación de la onda y otros fenómenos que intervienen en la transmisión de información entre uno y otro.
- b) **La derrota y localización** de las unidades de exploración (también llamadas observadores) con relación a la posición prevista y movimiento del objetivo (objeto de la exploración).
- c) **La dirección de despliegue** de las fuerzas navales, de acuerdo con las necesidades para enfrentarse con la amenaza de un modo eficaz, esto es alcanzar el mejor resultado posible con las fuerzas de que se dispone.

La ciencia de la detección, como una rama de la táctica naval, busca soluciones a los problemas de contacto y seguimiento de fuerzas hostiles o fuerzas que se presume no son amigas. Esta rama de la táctica alcanza su objetivo a través de la aplicación de la ingeniería, física, matemática y estadística. Sus conclusiones se establecen en términos de probabilidad, es decir, probabilidad de detección.

2.1.1 Generalidades Concernientes a la Detección

Se considera **descubierto** “un objetivo cuando nos damos cuenta de su presencia y conocemos su posible posición. Notemos que esta definición no considera la detección de objetos que no son de interés para el explorador. La adquisición subsiguiente de información del objetivo, se considera como **seguimiento**”.

Asimismo, es necesario comprender que si no desea información relativa a la ausencia o presencia de objetivo, es necesario una interacción directa o indirecta entre dicho objeto y el explorador. Esta interacción se da a través de un intercambio de energía entre el objeto y el explorador. Y es que, sencillamente, la naturaleza no tiene otro medio que el de transferir energía, gracias a la cual puede adquirirse la información a una distancia determinada. Es por esta razón por la que cualquier acción sobre la búsqueda sea por **sonar, radar o visual** implica una transferencia de energía.

Debe quedar muy claro que la detección de un objeto implica de manera indirecta o directa un intercambio de energía entre el objetivo y el “observador” o el que detecta al objetivo.

Existen dos categorías de sistemas de sensores para detectar objetivos y que son utilizados en guerras u operaciones navales, las cuales son:

- a) SONARES: instrumentos utilizados en la detección de objetos bajo el agua.
- b) RADARES: Instrumentos utilizados para objetos en superficies y en el aire.

Es importante mencionar que para el desarrollo del presente trabajo de investigación, únicamente se hará una descripción más a fondo del radar, por el tipo de problemática y objetivo que se describió anteriormente.

Un punto significativo del análisis con sensores para su aplicación en operaciones navales es el siguiente: **encontrar la probabilidad que un sensor determinado detecte un objetivo fijado en un instante dado bajo condiciones ambientales determinadas.** Tal probabilidad es denominada la probabilidad de detección de una sola mirada o instantánea. Aunque la probabilidad de detección pueda ser denominada instantánea, ésta puede contribuir en el tiempo e integrada para un intervalo finito posterior de tiempo. Este intervalo posterior es usualmente corto, de pocos segundos para un radar y desde 20 segundos hasta menos de 20 minutos para un sonar.

2.1.2 Hechos Básicos que se Consideran en todo Tipo de Búsqueda¹.

Los objetivos no deben estar demasiado lejos y la línea visual, desde el observador, no debe estar completamente tapada, para que pueda ser vista. El radar no revelará la presencia de objetivos, cuando las condiciones atmosféricas o el medio ambiente en que se reciben los ecos son adversas. En esencia, podemos decir que es necesario conocer las limitaciones intrínsecas del equipo o instrumento para detectar las condiciones externas que limitan aún más las posibilidades de detección del instrumento que se está utilizando.

Aún cuando las condiciones físicas hagan la posible detección, no debe esperarse ésta, inevitablemente. La detección es un acontecimiento que, bajo condiciones apropiadas, tiene una probabilidad positiva, que varía de cero a uno. Así cuando el objeto apenas empieza a satisfacer las condiciones físicas de su posible detección aumenta y puede llegar a ser prácticamente cierta. Sin embargo, la experiencia demuestra que, un objeto perfectamente visible puede no ser encontrado. Aviones en misiones de observación y reconocimiento en días claros y soleados, han pasado cerca de barcos grandes y sin embargo, no fueron detectados.

Debe tenerse en cuenta constantemente, que todo instrumento de detección se basa en una última consideración: el ser humano; sus éxitos dependen de su entrenamiento, estado de atención o cansancio. De hecho, aún en condiciones físicas fijas y constantes pueden tener lugar innumerables fluctuaciones. Por ejemplo, un eco radar cambia su aspecto de un momento a otro con el cabeceo y balanceo del barco; los alcances del sonar experimentan oscilaciones, ya que un objeto que puede no ser detectado en un cierto instante, puede serlo en un momento más tarde. El

¹ Daniel H. Wagner, W. Charles Mylander, “Naval Operations Análisis”, Naval Institute Press, Third Edition, United States of America, 1999.

reconocimiento de una señal emitida por un objeto y por lo tanto la probabilidad de su detección depende:

- a) De que la clase y características de la señal sea conocida por el observador.
- b) De la intensidad de la señal presentada al observador.
- c) De la intensidad de otras señales concurrentes conocidas como “ruidos”.

El observador que trata de localizar un objetivo por medios visuales, radar, sonar, etc. , estará utilizando algunos de los dos procedimientos básicos:

- a) El observador dirige una sucesión de “vistazos rápidos”. Un caso típico es el del eco del radar, el cual, en cada barrido proporciona un “vistazo”, teniendo lugar varias acciones de este tipo en un tiempo o intervalo específico.
- b) Cuando se fija la mira, es decir, el observador está “continuamente mirando”, fijando su “mirada” sobre un campo donde pueda estar situado el objetivo.

El caso del radar es intermedio, ya que su exploración es rotatoria, pertenecería al primer caso, pero si esta exploración rotatoria, es lo suficientemente rápida y hay una persistencia de imagen en la pantalla, puede ser considerada como de “vista continua”.

Del mismo modo, la detección visual por una antena que trabaja despacio en un ángulo grande, pertenece al primer caso más que al segundo. Muy a menudo la clasificación del método de detección depende simplemente, de la que proporcione la más cercana y conveniente aproximación.

Barridos Separados². Entiéndase como barrido, una exploración de un área en específico, con la finalidad de detectar un objetivo. Por tanto los barridos separados o discretos son exploraciones estadísticamente independientes una de la otra.

Barridos Separados Constantes. En el caso especial en que no cambien las condiciones durante la exploración, cada probabilidad de barrido tiene el mismo valor.

Barridos Continuos. Cuando se realiza una exploración de un área en específico con la finalidad de detectar un objetivo y se utiliza un detector de naturaleza continua (sistemas optrónicos), permite recibir la información de forma continua, por lo tanto, la detección será posible en cualquier instante durante la exploración.

2.2 Clasificación General de Búsqueda.

En el ámbito naval existen diferentes formas de realizar una búsqueda, esto dependiendo del tipo y características del objeto que se quiera detectar, por ejemplo: un buque de superficie, un avión, un submarino, etc. En la Figura 2.1 se esquematiza de forma general la clasificación de búsquedas que se pueden realizar.

² Esta definición es resultado de una investigación de campo.

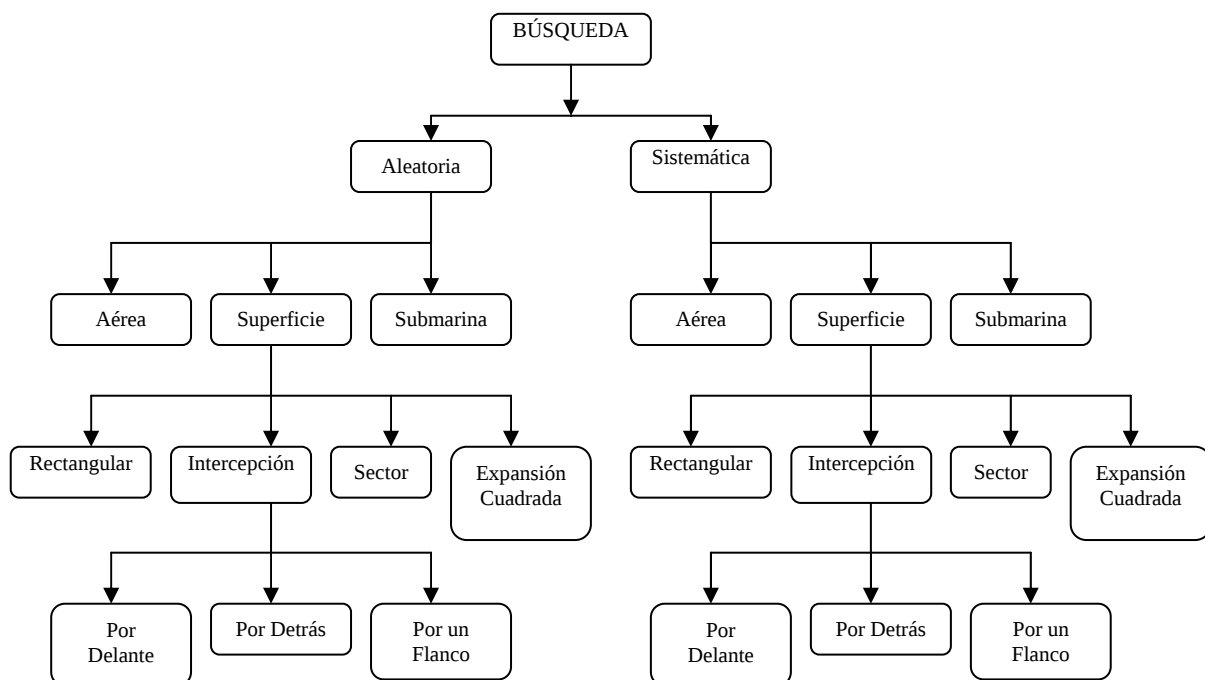


Figura 2.1

Búsqueda Aleatoria

Exploración de un área geográfica marítima con la finalidad de detectar objetivos o confirmar su existencia, en donde se asume que éste se encuentra en una posición aleatoria en dicha área. Este supuesto es frecuentemente válido ya que la situación de un objetivo no es usualmente conocida y el movimiento del objetivo es a lo más estimado. Considerando además que este objetivo tiene alguna posibilidad de ser detectado. De forma general, es aquella búsqueda que se realiza cuando se desconocen datos por completo del objetivo.

Búsqueda Sistemática

Exploración de un área geográfica marítima en donde se tiene cierta información de algunas de las características del objetivo a buscar o detectar, no se empieza este proceso de búsqueda con estimaciones, sino que empieza con la poca o mucha información recabada sobre el objetivo en estudio.

Búsqueda Aleatoria en Superficie

En una búsqueda de superficie en general el explorador sale en busca de un objetivo en la superficie de la mar. En este tipo de búsqueda los exploradores pueden ser: aviones y buques. Utilizando únicamente dos tipos de sensores para realizar dicha búsqueda y que son: los Radares y los Optrónicos.

Asimismo en la búsqueda de superficie existe una subclasificación, la cual es:

A) Búsqueda Rectangular

Es aquella búsqueda en donde se emplean aviones o buques de superficie; cuando una búsqueda a un radio máximo no es de importancia y la información sobre el propósito de la búsqueda es indefinida. Es una búsqueda económica y los exploradores pueden mantener posiciones relativas uno de otro, sin dificultad. En la búsqueda rectangular las derrotas de exploración son paralelas y cuando está presente más de un explorador la línea de exploración es normalmente en ángulo recto a las derrotas de exploración.

B) Búsqueda de Sector

Está destinada a ser empleada principalmente por aviones, para búsqueda a distancia máxima y frecuentemente se emplea para proporcionar la alarma más anticipada posible de un objetivo que se aproxima. No es económica debido a que proporciona una probabilidad de contacto inadecuadamente alta al principio y una probabilidad decreciente cerca de las extremidades, en donde frecuentemente se requiere de más exploración.

Cada avión vuela tres tramos; un tramo de salida, un tramo transversal y un tramo de entrada. El giro del tramo de salida hacia el tramo transversal es a una distancia que es menor al radio máximo e igual a la mitad de separación de derrotas calculadas en aquel punto. Antes de girar al tramo de entrada, el avión avanza sobre el tramo transversal una distancia igual a esta separación de derrotas.

C) Búsqueda de expansión cuadrada

Es empleada principalmente por aviones, especialmente cuando se conoce la posición del enemigo, pero no se tiene conocimiento de su rumbo y velocidad. Puesto que la búsqueda de expansión cuadrada cubre primero el área más probable de detección y luego las áreas exteriores menos probables, particularmente valiosa cuando un contacto anticipado es de consideración primaria. Debido al gran número de rumbos de corta duración durante la primera parte de la búsqueda y la consiguiente dificultad de mantener una derrota exacta sobre el terreno; la búsqueda de expansión cuadrada no es tan eficaz como una búsqueda rectangular correspondiente en el área. Está limitada a áreas relativamente pequeñas; puesto que es virtualmente imposible coordinar las actividades de dos o más aviones en aquella búsqueda.

D) Búsqueda por Delante, Detrás y Flanco

Esta destinada a un objetivo, cuya posición previa es conocida y cuyos límites de rumbo y velocidad pueden ser supuestos. Este tipo de búsqueda no es empleado por aviones, debido a que su elevado régimen de búsqueda de los aviones por lo general hace innecesario llevar a cabo los procedimientos detallados de una búsqueda de este tipo. Se basa en el supuesto que el objetivo mantiene rumbo y velocidad constante durante toda la búsqueda.

La elección de la búsqueda depende de la posición relativa de los exploradores y del objetivo. Las búsquedas por detrás y de flanco requieren que los exploradores tengan mayor velocidad que el objetivo.

Búsqueda por delante es aquella donde la línea de exploración se forma inicialmente sobre una línea en ángulo recto a la derrota más probable del objetivo y con su centro sobre esta derrota en el punto más avanzado del área probable del objetivo.

Búsqueda por detrás es aquella en donde la línea de exploración se forma inicialmente sobre una línea en ángulo recto a la derrotas más probable del objetivo y con su centro sobre esta derrota y atrás del punto menos avanzado del área probable del objetivo.

Búsqueda de flanco es aquella que puede ser efectuada por uno o dos métodos. Búsqueda alejándose de la posición informada del objetivo y búsqueda hacia la posición del objetivo.

De forma general y específica de todos estos tipos de búsqueda se da en la **Tabla 2.1** .

La selección del tipo de búsqueda se dará en consideración al tipo de área a explorar, al tipo de barco y al ancho de barrido³ del radar. Asimismo la determinación exacta de la forma y tamaño del área es esencial para el éxito de la búsqueda y es el primer paso en el planeamiento de la misma.

³ Entendiéndose como **Ancho de barrido**, al parámetro que permite medir la capacidad de detección de un sensor en términos de distancia y se representa por **W**. Este concepto establece que cada equipo de detección que rastrea a través de un área, efectivamente barre una anchura. Varía con el medio de búsqueda empleado, la naturaleza y posición del objeto buscado, la velocidad de la búsqueda, las condiciones meteorológicas, la falibilidad humana y del material así como otras variables.

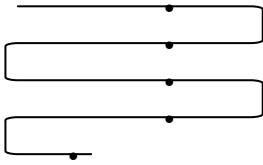
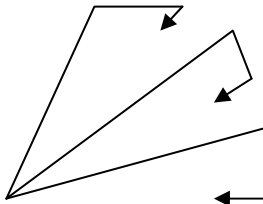
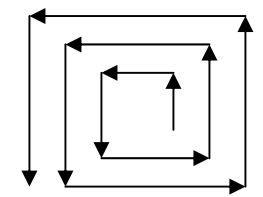
TIPOS BÁSICOS DE BUSQUEDAS				
TIPOS DE BUSQUEDAS	FORMA ESQUEMATICA	TIPOS DE EXPLORADORES	CONDICIONES PARA LA SELECCIÓN	VENTAJAS
				DESVENTAJAS
BÚSQUEDA RECTANGULAR Una serie de barridos simultáneos o sucesivos a través del área designada.		Aviones, buques de superficie, submarinos o una combinación de estos.	1. El conocimiento de objetivo posibles esta limitado a posición aproximada. 2. La búsqueda al radio máximo no es una consideración.	1. Económica 2. Los exploradores pueden mantener posición relativa entre si sin dificultad.
				1. Los objetivo en movimiento pueden evitar los barridos sucesivos de un explorador
BÚSQUEDA DE SECTOR Una búsqueda radial de una parte o toda el área circular		Generalmente aviones	1. La búsqueda a la máxima distancia es requerida para proporcionar la alarma más anticipada posibles por la aproximación del enemigo. 2. Se busca hasta una distancia en un sector o ruta especificada de la posible aproximación del enemigo.	1. Más efectiva en las primeras etapas con recubrimiento.
				1. No es económico 2. Los claros en los extremos pueden permitir evasión.
BÚSQUEDA DE EXPANSIÓN CUADRADA Un diagrama en el cual una búsqueda rectangular se expande hacia fuera desde un punto		Generalmente aviones pero pueden usarse buques	1. Se conoce la posición aproximada de un objetivo a una hora y el movimiento del objetivo es impredecible. 2. El objetivo debe ser ubicado después que el explorador ha llegado a su última posición informada.	1. Cubre primero el área más probable.
				1. Limitada a áreas pequeñas 2. El mantenimiento de estación es difícil, generalmente limitada al uso de un explorador.
BÚSQUEDA DE INTERCEPCIÓN Una forma especial de búsqueda de área usado cuando es posible reducir el área a ser cubierta.		Normalmente buques de superficie	1. Se conocen la posición previa rumbo y velocidad del enemigo dentro de límites razonables de exactitud.	1. Economía en el esfuerzo de búsqueda.
				1. Se necesita procedimiento en detalle. 2. El éxito depende del rumbo y velocidad que se supone mantiene el enemigo dentro de los limites.

Tabla 2.1

Capítulo III

La Decisión

De los métodos que se usen, su objetivo es dar una base cuantitativa al directivo para tomar una decisión. Pero no trata de relevar al directivo de la responsabilidad de tomar una decisión sino, solamente, de ayudarlo a sopesar con mayor precisión, el valor relativo de cada alternativa para que pueda tener en cuenta, tanto los aspectos subjetivos como los cuantitativos.

El proporcionar al directivo una base cuantitativa, no significa hacerle más fácil el proceso de la decisión, ya que, en muchos casos, lo hace realmente más difícil, la ventaja real, es que, el directivo conoce con mayor precisión, las consecuencias que tendrá su decisión.

3.0 Grados de Riesgo

Las decisiones se deberán tomar con varios grados de conocimiento sobre las condiciones, bajo la cuales, se desarrollará una operación. Las alternativas que se muestran al directivo representan:

Estrategias o alternativas, las cuales son diferentes planes de exploración.

Estados de la naturaleza, los cuales son los diferentes estados de la mar.

Los resultados, ya que estos son las probabilidades de detectar el objeto que se busca.

Claramente se observa que, la cantidad de riesgo envuelto en la decisión de tomar una alternativa, queda determinado por la incertidumbre del estado de la mar existente durante la operación, de la cual se desprenden los siguientes casos:

Decisión bajo certeza, cuando el estado de la mar es conocido, buscar simplemente el mejor resultado.

Decisión bajo riesgo, cuando se desconoce el estado de la naturaleza, pero se conoce en cambio la probabilidad de cada uno de ellos. Tomando una decisión bajo riesgo, se debe escoger la estrategia que optimice el resultado esperado de la MOE.

Decisión bajo incertidumbre, el directivo no conoce las probabilidades con que se presentan los diferentes estados de la naturaleza, en donde se sugieren diversos criterios.

Teoría de juegos, sucede cuando el estado de naturaleza está controlado por un oponente racional, de quien se puede esperar que escoja aquel que frustre el objetivo del directivo. Los problemas de la decisión de este tipo se llaman juego de estrategia y son objeto de estudio en una rama matemática altamente desarrollada, llamada, teoría de juegos.

3.1 Criterios Bajo Incertidumbre

Primer criterio; **Pesimista o Criterio de Wald** (llamado así por que fue Abraham Wald quien lo expuso), minimiza el riesgo intrínseco de tomar una decisión. Se le llama también **criterio maximim**, ya que, se halla primero el resultado mínimo de cada opción y después se escoge la opción que otorga el máximo valor de estos resultados mínimos garantizados.

El segundo criterio es similar al primero, pero rechaza la idea pesimista. Fue presentado por Leonid Hurwicz quien sugirió que, todo directivo tiene algo de optimismo en una situación determinada. Este **Criterio maximax** equivale a presumir que la naturaleza es benevolente y, por eso, se encontrará en la condición mas favorable para el directivo. Por el contrario, un pesimista completo escogerá el criterio maximim descrito previamente; que equivale a presumir, que la naturaleza es malévola y tendría el peor resultado, que es lo que frustrará en mayor grado la decisión del directivo.

Hurwicz sugiere al directivo, que ponga un grado de optimismo en una escala de 0 a 1, donde el cero representa un pesimismo completo y el 1 un optimismo completo.

El tercer criterio se basa en la tendencia de algunos directivos de revisar su decisión, una vez que se ha ejecutado la acción, con el fin de ver de que forma se habría mejorado el resultado si se hubiese previsto, correctamente, el estado naturaleza. Esto proporciona la posibilidad de establecer un nuevo criterio para la decisión: el de minimizar el **Criterio de riesgo de error**. Este criterio fue establecido por Leonard Savage, quien sugiere, que el criterio de Wald se aplique al criterio de riesgo error con el fin de determinar la estrategia que garantice el menor riesgo de error.

El criterio final, llamado **Criterio de racionalidad** se atribuye a Laplace. El argumentaba que la completa incertidumbre sobre el probable estado de la naturaleza, es equivalente a decir que todos son igualmente probables. Esto es si cualquier estado se presume más probable que otro será a causa de que se tiene más información y, por lo tanto, el directivo no se enfrentara a una incertidumbre completa. Las implicaciones, y la validez, de este argumento se han discutido mucho en el pasado y, seguramente, se continuará discutiendo por muchos años.

3.2 Métodos de Probabilidad de Búsqueda y Detección¹

Curva de Alcance Lateral. Es una representación gráfica de la probabilidad de detección de un objetivo, el cual pasa a cualquier alcance lateral del radar.

Raramente el movimiento relativo del objetivo será directamente hacia el detector. Generalmente, el objetivo se moverá a lo largo de una línea que atraviesa la zona de posible detección del detector. La distancia mínima del objetivo (DMO) al equipo se ha definido como **alcance lateral**, con respecto a un equipo de detección. El alcance lateral, es un parámetro físico normalmente representado por X.

¹ Daniel H. Wagner, W. Charles Mylander, "Naval Operations Analysis", Naval Institute Press, Third Edition, United States of America, 1999.

La región o zona de posible detección, es al interior de un círculo centrado sobre el equipo de detección, que tiene un radio igual a la máxima distancia posible de detección R_m . Como se ve en la **Figura 3.1**

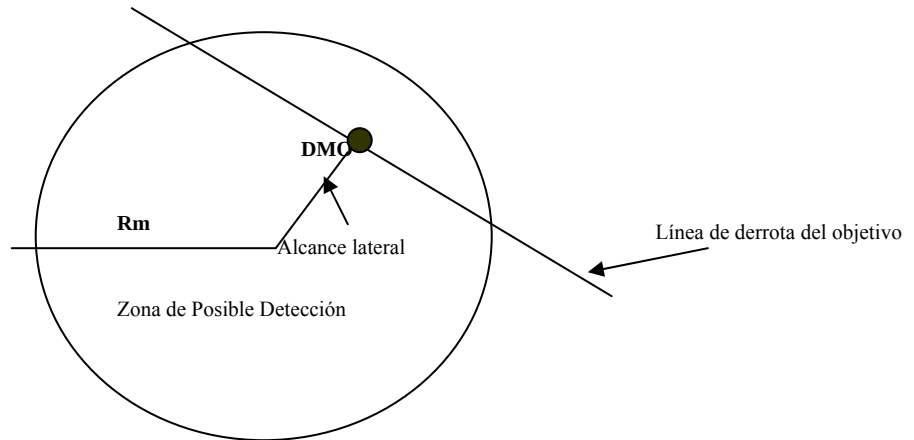


Figura 3.1

Para un conjunto fijo de detecciones ambientales, la probabilidad acumulada de detectar un objetivo se incrementa durante el tiempo que el objetivo permanece en la zona.

Esta probabilidad acumulada se expresa como $P(x)$. La representación gráfica de $P(x)$, para todos los valores de x , se le conoce como una curva de alcance lateral típica que se representa en la **Figura 3.2**

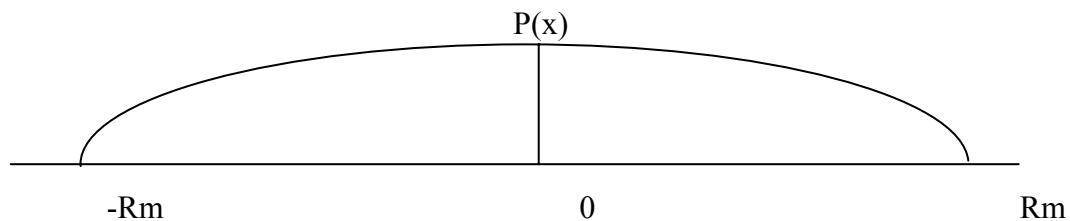


Figura 3.2

Esta curva representa la probabilidad acumulada de detección para un objetivo en particular, bajo un conjunto particular de circunstancias ambientales, con un equipo de detección determinado. Cualquier cambio de las condiciones supondrá una curva de alcance lateral diferente, por lo tanto, cada equipo de detección tendrá una familia de curvas de alcance lateral, cada una de las cuales corresponderá a un tipo de objetivo y una variedad de condiciones ambientales.

Detección de Blancos Aleatoriamente Distribuidos. Considérese el caso de la búsqueda de un objetivo, en el cual se supone que el objetivo está situado aleatoriamente, en un área que están barriendo los quipos de detección. Consideremos que este objetivo tiene una probabilidad de ser detectado, esto es, su alcance lateral tiene un valor entre $-R_m$ y R_m . Este objetivo es tan probable que siga una línea de derrota relativa a través de la zona de detección, a una distancia

lateral dada como a otro cualquier alcance lateral. Matemáticamente, esto significa que si la variable aleatoria X tiene una distribución uniforme de probabilidad para todos los valores comprendidos entre $-R_m$ y R_m , por lo tanto su función de densidad de la variable x es.

$$\dots\dots\dots (1).$$

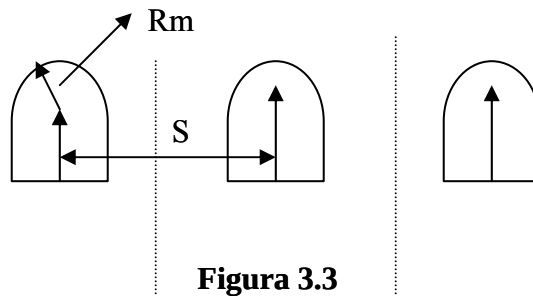
Ahora, recordemos la función $P(x)$, que nos da la probabilidad acumulada de detectar un objetivo que pasa a un alcance lateral específico x . la probabilidad de detectar un blanco cuya distancia de alcance lateral se desconoce es el valor esperado de $P(x)$, donde x puede tomar todos los valores posibles de alcance lateral.

$$E [P(x)] = \int_{\text{para toda } x} P(x) f(x) dx \quad \dots\dots\dots (2).$$

En este caso, consideremos que x está uniformemente distribuido entre $-R_m$ y R_m , el valor esperado será:

$$E [P(x)] = 1 / 2 R_m \int_{-R_m}^{R_m} P(x) f(x) dx \quad \dots\dots\dots (3).$$

Lo cual da la probabilidad media esperada de detección de un objetivo que atraviesa aleatoriamente la zona de posible detección. En la siguiente **Figura 3.3** se representa un caso genera.



En este caso, varias unidades exploradoras llevan a cabo un plan de exploración separada a una cierta distancia S (separación entre derrotas), de tal modo que no existen zonas traslapadas de detección, por lo tanto, $S > 2 R_m$.

Llamaremos x a la distancia del objetivo al explorador más cercano, entonces el valor de x está uniformemente distribuido entre $-S/2$ y $S/2$ y $f(x)$ tiene el valor de $1/S$. por lo que la probabilidad de detección esperada por un explorador sería, considerando la ecuación (3) anterior de la esperanza matemática:

$$E [P(x)] = 1 / S \int_{-R_m}^{R_m} P(x) f(x) dx \quad \dots\dots (4)$$

Ancho de Barrido. Es un parámetro que nos permite medir la capacidad de detección de un sensor en términos de distancia, este concepto es conocido como ancho de barrido y se representa por W. Este concepto establece que cada equipo de detección que rastrea a través de un área, efectivamente barre una anchura.

$$P(\text{Detección}) = \text{Ancho de barrido} / S = E [P(x)] = 1 / S \int_{-R_m}^{R_m} P(x) dx \quad \dots(5)$$

Si todos los blancos dentro del ancho de barrido, fuesen detectados y no fuesen los que se encuentran en le área, entonces, la probabilidad de detección de un solo objetivo sería precisamente, la fracción de todos los objetivos dentro del ancho de barrido, para este caso particular es:

Por lo tanto, para ser real el ancho de barrido se debe definir en magnitud, al igual que el área, W de la curva de alcance lateral. El símbolo W representa entonces, físicamente, el ancho efectivo de la zona de detección del instrumento. Su valor se halla calculando el área de la curva de alcance lateral. La probabilidad promedio de detección del objetivo distribuido aleatoriamente por medio de instrumentos de detección, cuyas zonas de detección no están traslapadas es:

$$P(\text{Detección}) = \frac{\int_{-R_m}^{R_m} P(x) dx}{S} \quad \dots(6)$$

Búsqueda Aleatoria. La búsqueda de un objetivo asume que éste se encuentra en una posición aleatoria en un área marítima. Este supuesto es frecuentemente válido ya que la situación de un blanco no es usualmente conocida y el movimiento del objetivo es a lo más estimado. Considerando además que el objetivo tiene alguna posibilidad de ser detectado.

Si consideramos como un supuesto: que tenemos un área geográfica marítima A, un buque como objetivo dentro de esa área marítima; así mismo que este objetivo tiene la misma probabilidad de estar en cualquier punto del área y que el sensor de la unidad desarrolla una ruta aleatoria dentro de esa área. Supongamos también que el observador explora el área sin seguir un plan sistemático o metódico. En esta búsqueda aleatoria, ¿Cuál es la probabilidad de que la detección tenga lugar, en el tiempo en el que el observador navega L millas, dentro de esta área.?

Sea P(x), la curva de alcance lateral del observador, con este objetivo concreto en el ambiente existente. Dividamos la derrota del observador en N segmentos de igual longitud L / N, que tenga una forma aproximada a la línea recta, como se muestra en la siguiente **Figura 3.4** .

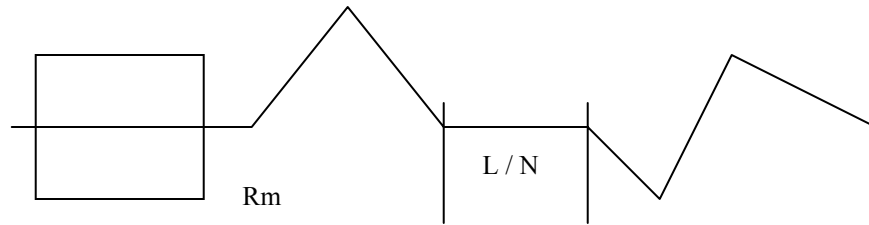


Figura 3.4

Para que la detección ocurra en el primer segmento deben tener lugar dos circunstancias. Sea B, el caso en que el objetivo está dentro del área de longitud L / N y anchura $2R_m$ (esta área la designaremos por AN) de tal modo que exista alguna probabilidad de detección, y sea C, el caso en que el objetivo se detecta.

Entonces de la teoría clásica de probabilidad (ver Anexo A):

$$P(B) = AN / A = (2 R_m L / N) / A = (2 R_m L) / NA \quad \dots\dots(7)$$

De la teoría de probabilidad condicional (ver Anexo A) , la probabilidad de que el objetivo sea detectado dado que se encuentra dentro del área de longitud L / N :

$$\begin{aligned} P(c / B) &= P(c \cap B) / P(B) \\ P(C / B) &= P(B) P(C / B) \end{aligned} \quad \dots\dots(8)$$

El evento C es el caso cuando el blanco es detectado, el cual únicamente se realizará con la condición de que esté en B, es decir, que este dentro del área de longitud L / N y anchura $2 R_m$, por lo tanto, B está contenido en C y : $P(C \cap B) = P(C)$.

$$\text{Por lo tanto: } P(C \cap B) = P(C) = P(B) P(C / B) \quad \dots\dots(9)$$

el significado de $P(C / B)$ es el valor esperado o valor medio de la probabilidad de detección en un objetivo distribuido aleatoriamente.

$$\text{Es decir } P(C / B) = E [P(x)] = \frac{1}{2 R_m} \int_{-R_m}^{R_m} P(x) dx \quad \dots\dots\dots(10)$$

Por lo tanto:

$$P(C \cap B) = P(C) = P(B) P(C / B) = \frac{2 R_m L}{NA} \frac{1}{2 R_m} \int_{-R_m}^{R_m} P(x) dx = \frac{L}{NA} \int_{-R_m}^{R_m} P(x) dx \quad \dots\dots\dots(11)$$

Y como el ancho de barrido es

$$W = \int_{-R_m}^{R_m} P(x) dx \quad \dots\dots\dots(12)$$

El área bajo la curva de alcance lateral puede expresarse como sigue:

$$P(C) = WL / NA \dots\dots\dots(13)$$

Y la probabilidad de que la detección no tenga lugar durante una búsqueda aleatoria (en un segmento) sería entonces:

$$P (\text{no detección}) = [1 - (WL / NA)] \dots\dots\dots(14)$$

Admitiendo independencia en los distintos n eventos, dado que son N segmentos:

$$P (\text{no detección}) = [1 - (WL / NA)]^n \dots\dots\dots(15)$$

Bajo la misma suposición de independencia de los eventos y dado que la función de efectividad de la búsqueda mide la probabilidad de detección, entonces:

$$P (\text{no detección}) = 1 - [1 - (WL / NA)]^n \dots\dots\dots(16)$$

Adicionalmente, podemos realizar la siguiente sustitución:

De la ecuación (13) tenemos que: $P(C) = WL / NA$ que es la probabilidad de detección del objetivo, por lo tanto la probabilidad de no detección sería:

$1 - P(C) = 1 - WL / NA$; suponiendo los N segmentos y la independencia de cada uno de los eventos:

$$[1 - (WL / NA)]^n = P(\text{no detección}) \dots\dots\dots(17)$$

Realizando algunos trucos matemáticos, tenemos que:

$$[1 - (WL / NA)]^n = e^{N \ln [1 - (WL / NA)]}, \text{ observe que no se ha hecho nada.}$$

Si hacemos la consideración que WL / NA es muy pequeña, tenemos que:

$$\ln [1 - WL / NA] = - WL / NA \dots\dots\dots(18),$$

por lo tanto:

$$[1 - (WL / NA)]^n = e^{N \ln [1 - (WL / NA)]} = e^{N [- WL / NA]} \dots\dots\dots(19)$$

Sustituyendo la ecuación anterior en (16) podemos tener que:

$$P (\text{Detección}) = 1 - [1 - (WL / NA)]^n$$

$$P (\text{Detección}) = [1 - e^{N [- WL / NA]}]$$

$$P (\text{Detección}) = [1 - e^{-WL / NA}] \dots\dots\dots(20)$$

Con tal que la fracción cubierta del área efectiva de cada segmento sea pequeña.

La ecuación (20) puede ser expresada en términos del ancho de barrido, resultando:

$$P(\text{Detección}) = 1 - e^{-W/S} \quad \dots\dots(21)$$

A la cantidad W / S se le conoce como factor de barrido.

La fórmula final de la probabilidad de detección de una búsqueda aleatoria de un avión, considerando la expresión (20), es:

$$P\{\text{Detección}\} = 1 - e^{-WVT/A} \quad \dots\dots(22)$$

Al analizar la fórmula final de la probabilidad de detección, se observa que la probabilidad de detectar a un objetivo está en función del ancho de barrido, la velocidad de plataforma que monta al sensor y del tiempo t , utilizado en la búsqueda considerando un área A determinada.

El análisis desarrollado anteriormente está basado en las siguientes consideraciones que siempre deben ser tomadas en cuenta:

- a) La posición del objetivo esta aleatoriamente distribuida en A
- b) La exploración se realiza de una forma aleatoria, tal que sus segmentos individuales pueden ser considerados independientes.
- c) $2R_m$ es pequeño comparado con L / N , de tal forma que pueden despreciarse las áreas no barridas y las traslapadas de cada segmento.

La importancia de este tipo de búsqueda no es que presente un tipo particular de exploración operacional, sino que representa una exploración teórica, en la cual se tiene la mínima información del blanco y no se emplea una exploración sistemática. De aquí que, en el caso en que se sepa algo más del objetivo y se emplee una exploración sistemática, el mismo esfuerzo proporcionará una probabilidad de detección más alta.

Ley Inversa del Cubo. Es aplicable cuando se requiere calcular la probabilidad de detección; donde las curvas de posible alcance lateral y las distancias entre derrotas son diferentes. Asimismo cuando se conoce el ancho de barrido del sensor.

En general, este método particularmente ayuda a predecir la probabilidad de detección por medio de barridos paralelos y cuando se conoce el ancho de barrido que tienen los sensores de detección. Además de que es el mejor predictor teórico de resultados reales. Y la forma de calcularlo es a través de lo siguiente:

$$P_d = 2 \int_0^Z \varphi(t) dt \quad \dots\dots(23)$$

Donde ϕ es la función de densidad de probabilidad normal estandarizada con media cero y variancia 1.

Y además:

$$C = \sqrt{\frac{\Pi}{2}} \left[\frac{W}{S} \right] \quad \text{o} \quad C = 1.253 (W / S) \dots\dots\dots (24)$$

De donde C será el Factor de Barrido y se utilizan las tablas de probabilidad normal para su cálculo.

Capítulo IV

Aplicación de Búsqueda y Detección en Superficie

En este capítulo se analizarán y resolverán problemas prácticos y simples de la vida cotidiana; aplicando los métodos de búsqueda y detección antes mencionados, con la finalidad de demostrar la aplicación de dichos métodos en una gran diversidad de campos.

Cabe mencionar que estos mismos métodos de búsqueda pueden aplicarse a problemas navales de búsqueda y detección en superficie.

4.0 Problemas de Aplicación y Simulación

Los siguientes problemas de aplicaciones son resueltos analíticamente y simulados en el despliegue para su visualización, el despliegue se realizó en el lenguaje Matlab 7.0, bajo plataforma Windows XP, debido a facilidad en el uso de tablas de probabilidad, las cuales sirvieron para realizar cálculos en la programación, así como la ventaja de tener una herramienta para interfaces gráficas, las cuales son de mejor entendimiento para el usuario. Estas simulaciones describen, una a una cada búsqueda descrita anteriormente para cualquier situación y varían conforme a datos ingresados, cabe mencionar que no se genera base de datos, solo muestra resultados al instante, sin ser histórico.

En conjunto con Matlab 7.0 se desarrollo una simulación gráfica de cada una de las búsquedas aquí descritas, dichas simulaciones gráficas fueron hechas en Macromedia Flash 8.0 bajo plataforma Windows XP, dichas simulaciones son una representación fija de cómo es la trayectoria de las búsquedas, es decir no varían conforme se cambian datos para la búsqueda.

4.0.1 Búsqueda Rectangular

Planteamiento del problema

Realizando pruebas de calidad, una empresa decide poner a prueba sus detectores de metales.

La prueba consistió en dejar al azar una muestra del metal en un cuarto y realizando una búsqueda rectangular fijaron una determinada separación entre cada línea de búsqueda. Con la finalidad de ver el verdadero alcance de sus detectores de metales y que la eficiencia en la detección del metal al aplicar la búsqueda rectangular, de una confianza del 93% de éxito.

Datos:

Cuarto en donde se dejó el metal al azar: 12 x 9 [m]

Probabilidad de encontrar el metal : 93%

Ancho de Barrido (W) del detector = 1.5[m]

Velocidad de Búsqueda: 30 m/seg

Número de detectores: 4

Se requiere:

Derrota (S) = ?

Distancia recorrida por cada explorador?

Tiempo de búsqueda?

Solución:

1er. Paso:

Por medio de tablas de la normal acumulada y $P = 93\%$ se calculara C y S, en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

Donde C es el factor de barrido

$$Pd = 2 \int_0^z \phi(t) dt$$

$$0.93 = 2 \int_0^z \phi(t) dt$$

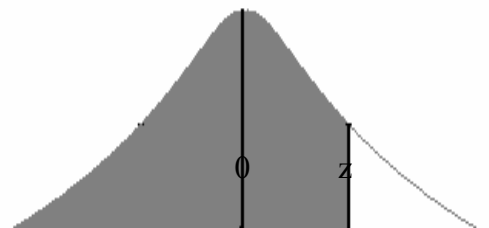
$$\frac{0.93}{2} = \int_0^z \phi(t) dt$$

$$0.465 = \int_0^z \phi(t) dt$$

Gráficamente se tiene



$-\infty$



$$\int_{-\infty}^z \Phi(t)dt = \int_{-\infty}^0 \Phi(t)dt + \int_0^z \Phi(t)dt$$

$$\int_{-\infty}^z \Phi(t)dt = 0.5 + 0.465 = 0.965$$

Ahora el valor de 0.965 se busca en tablas de la normal resultando una $C = 1.81$.

Nota: Es importante mencionar que en este cálculo tomamos una C normal teórica.

Y aplicando la ecuación (24) del Capítulo III,

$$C = 1.253 (W / S)$$

$$S = (1.253)[W / C] = (1.253)[1.5 / 1.81] = 1.038 = 1$$

2do. Paso:

Se calcula el número de barridos

a) Giros simultáneos

N = número de detectores

$$N^{\circ} \text{ Barridos} = \frac{\text{Longitud del lado menor}}{N \ S} = \frac{9}{4(1)} = 2.25 \text{ barridos}$$

b) Giros de búsqueda

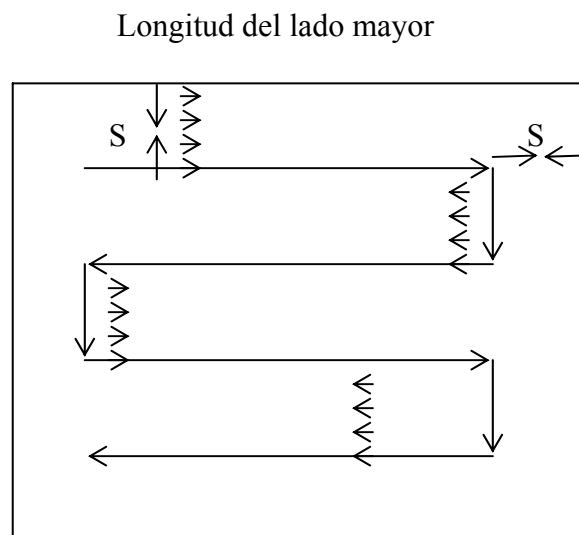
Igual que en giros simultáneos

3er. Paso:

Se calcula la distancia recorrida por los detectores.

a) Giros simultáneos

La búsqueda rectangular en giros simultáneos, sigue un tipo de trayectoria como lo podemos apreciar en la gráfica 4.0.1(a)



Gráfica 4.0.1(a)

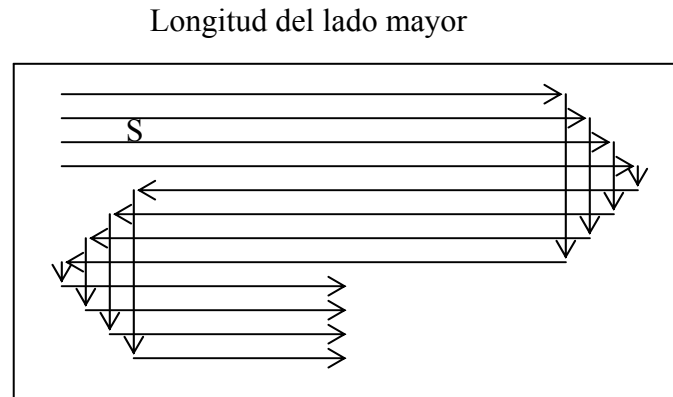
De esa forma la distancia recorrida, será la distancia total de la trayectoria (gráfica 4.0.1(a)), calculada de la siguiente manera:

$$\text{Dist. Recorrida} = N^{\circ} \text{ Barridos}(\text{Log. Lado mayor} - S) + (N^{\circ} \text{ Barridos} - 1)(N \cdot S) + S / 2$$

$$\text{Dist. Recorrida} = 2.25 (12 - 1) + (2.25 - 1) (1 \times 1) + 1 / 2 = 30.25 \text{ m.}$$

b) Giros de Búsqueda

La búsqueda rectangular en giros búsqueda, sigue un tipo de trayectoria en específico, como lo podemos apreciar en la gráfica 4.0.1(b)



Gráfica 4.0.1(b)

Por lo que la distancia recorrida, será la distancia total de esa trayectoria (Gráfica 4.0.1(b)), calculada de la siguiente manera:

$$\text{Dist. Recorrida} = \text{N}^\circ \text{ Barridos (Log. Lado mayor)} - S / 2$$

$$\text{Dist. Recorrida} = 2.25 (12) - 1 / 2 = 26.5 \text{ m.}$$

4to. Paso:

Se calcula el tiempo que dura la búsqueda, con velocidad constante, por lo tanto:

$$\text{Tiempo} = \frac{\text{Distancia}}{\text{Velocidad}}$$

a) Giro simultaneo

$$T = \frac{30.25}{30} = 1 \text{ hrs.}$$

b) Giro de búsqueda

$$T = \frac{26.5}{30} = 0.88 \text{ hrs.}$$

Los resultados del problema anterior, el cual se resolvió analíticamente; será resuelto mediante la aplicación desarrollada en esta tesis. Con el fin de mostrar la funcionalidad del sistema.

Al inicio del Sistema, se tiene una pantalla denominada menú; la cual muestra todos los tipos de búsqueda para superficie existentes. Misma que se utilizara para cada uno de los problemas que se desarrollaran en este capítulo, mostrada en la Figura 4.0.1(c).

Es importante mencionar que el sistema de aplicación fue desarrollado en Matlab versión 7.0 sobre plataforma Windows XP.

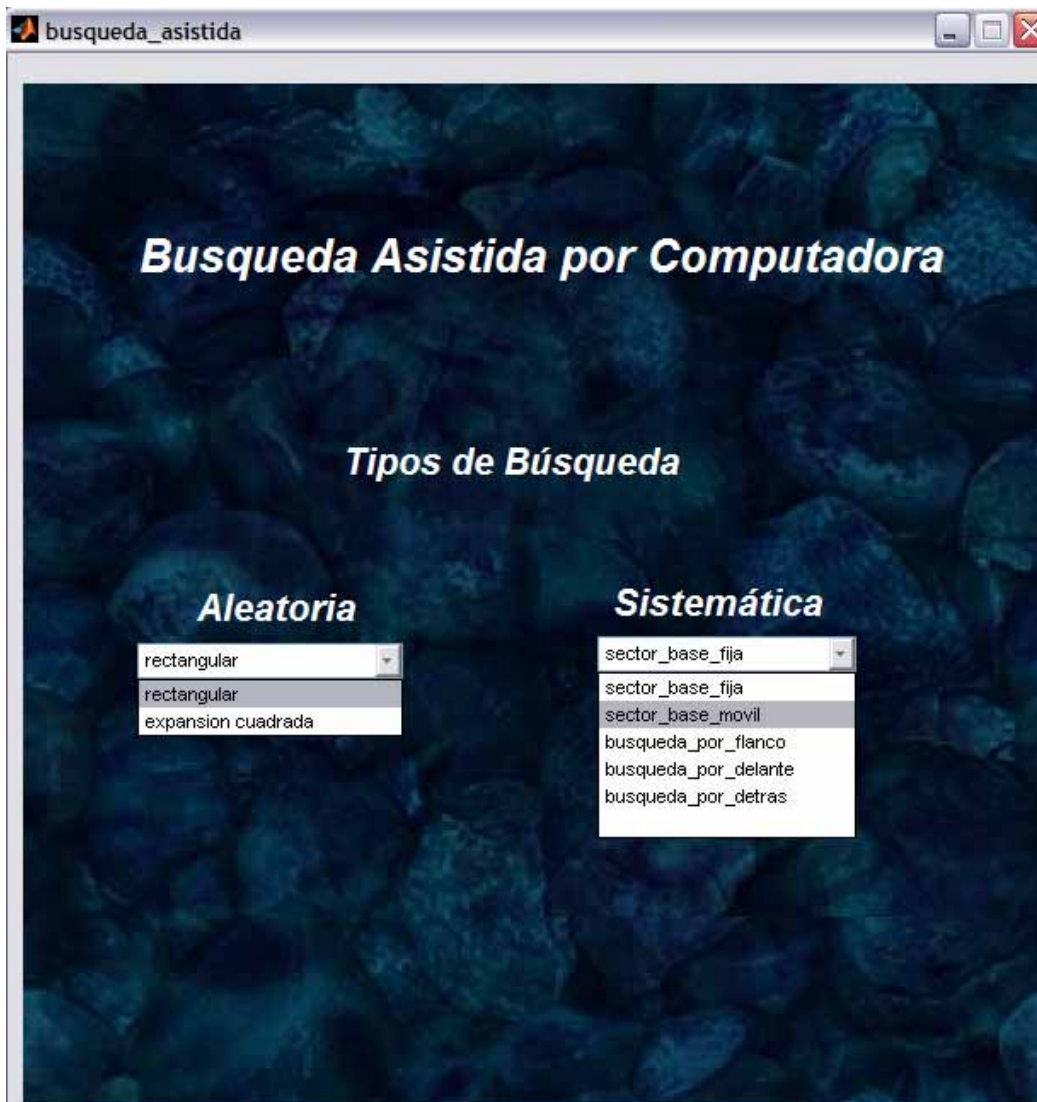


Figura 4.0.1(a) Menú principal del sistema

Con el menú de la Figura 4.0.1(a) se podrá seleccionar el tipo de búsqueda que se desee utilizar, dependiendo el tipo de problema a resolver.

Resolviendo el problema de búsqueda rectangular con el sistema desarrollado; se introducirán los datos requeridos para encontrar su solución.

La Figura 4.0.1(b), muestra la pantalla de simulación de Búsqueda Rectangular. La cual pide los siguientes datos:

- Ancho de barrido
- Probabilidad y Factor de Barrido: estos datos van de la mano para establecer un factor de barrido C, dada la probabilidad (P).
- Latitud y Altitud: De gran utilidad para establecer el área de búsqueda.
- Velocidad descrita por los buscadores.
- Número de exploradores que intervienen en la búsqueda.

rectangular

Anota los siguientes datos

Ancho de barrido (millas)	Area de busqueda	Latitud	Velocidad (Nudos)
2	12	30	
Probabilidad [0-100] %		Altitud	Numero de Exploradores
93	9	4	

Según tablas de la normal
 el factor de barrido C =
 con una probabilidad del:

Si desea usar el [C] obtenido, anotelo: en caso contrario deme su valor de C:

Resultados Teóricos

Derrota (S) Millas	Número de Barridos
1	Giros Simultaneos 2.25 Barridos
	Giros de Búsqueda 2.25 Barridos

Distancia Recorrida por los Exploradores	Tiempo que Dura la Búsqueda
Giros Simultaneos 30.25 Millas	Giros Simultaneos 1.00833 Hrs
Giros De Búsqueda 26.5 Millas	Giros De Búsqueda 0.883333 Hrs

Figura 4.0.1(b)

Al dar clic en el “**factor de barrido C**”; se obtendrá éste. El sistema cuenta con un archivo PDF para consultar la distribución normal, haciendo clic en el botón “**consultar tabla**”, de tal forma que permite verificar el factor de barrido calculado.

Posteriormente al dar clic en el botón **“BUSCAR”**, se genera automáticamente la siguiente información:

- Derrota:
- El número de barridos que harán los exploradores para los tipos de giros.
- Distancia que recorren los exploradores, para cada tipo de giro.
- Tiempo que dura la búsqueda para cada tipo de giro.

Este mismo procedimiento se llevará a cabo en todas las pantallas del sistema para cada uno de los tipos de búsqueda.

Después de observar resultados; al dar clic en **Simular**; el sistema mostrara una simulación gráfica de cómo es la búsqueda rectangular, en sus distintos tipos de giros, los cuales se mostraran dando un enter para su visualización respectivamente.

En la Figura 4.0.1(c) se muestran los giros Independientes; abarcando toda el área a cubrir, pero con la particularidad de que dependiendo del número de buscadores, el área se seccionará y hará para cada buscador una trayectoria de búsqueda rectangular.



Figura 4.0.1(c) Giros Independientes

La Figura 4.0.1(d) describe la trayectoria de búsqueda rectangular, pero ahora con giros simultáneos en donde todos los buscadores involucrados estarán paralelos uno del otro y al llegar

al limite del área, su giro será simultaneo uno tras otro para que el siguiente barrido que hagan también sea paralelo.

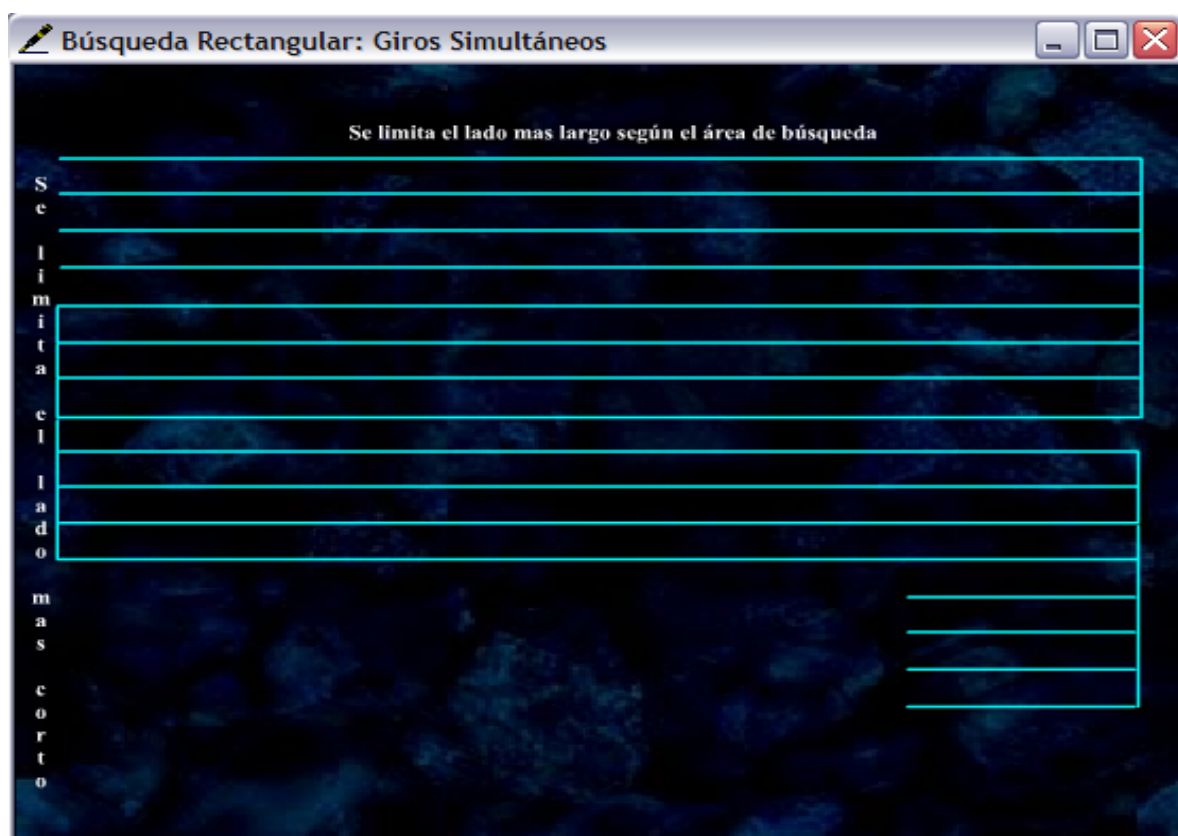


Figura 4.0.1(d) Giros Simultáneos

La Figura 4.0.1(e) describe la trayectoria de búsqueda rectangular, en donde todos los buscadores involucrados estarán paralelos uno del otro y al llegar al límite del área su giro será desfasado uno del otro para que al siguiente barrido que hagan también sea paralelo.

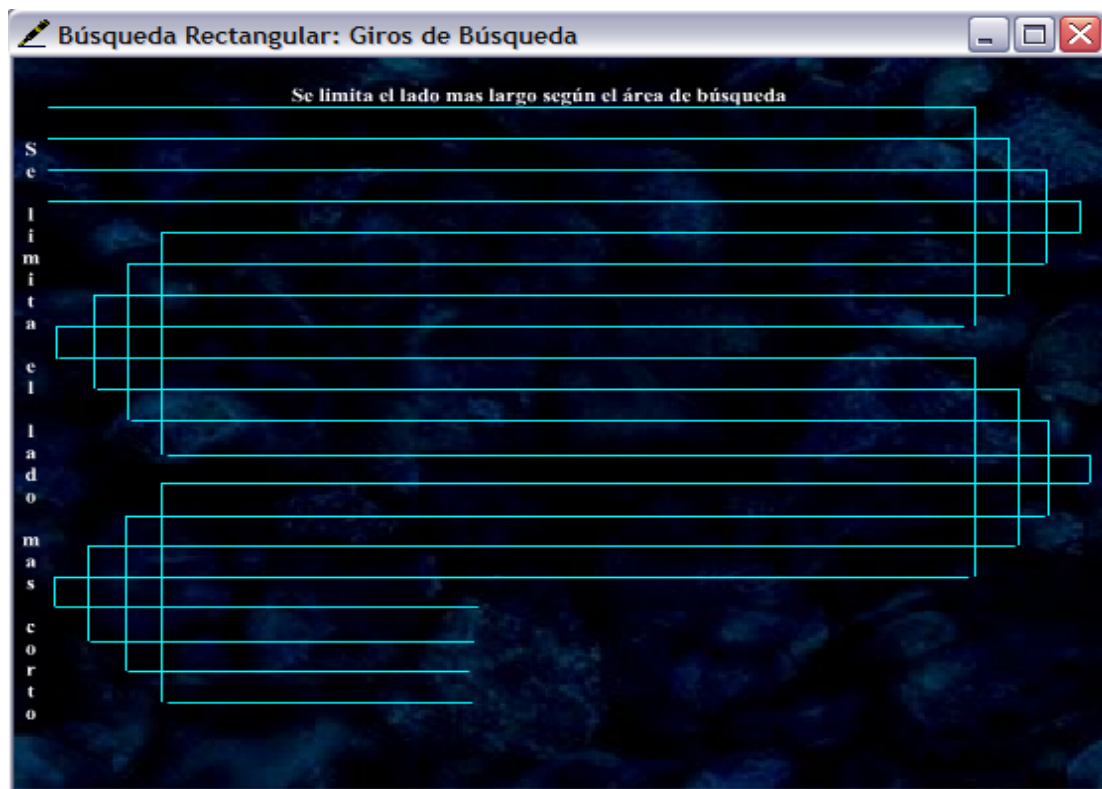


Figura 4.0.1(e) Giros de Búsqueda

4.0.2 Búsqueda de Expansión Cuadrada

Planteamiento del problema

En un balneario se organizó un concurso; en el que arrojaban a una alberca de 12 X 12 m, un objeto flotante al azar y aquel que encontrara el objeto en el menor tiempo posible era el ganador.

Muchos lo intentaron pero hubo uno en particular que mostró una técnica exitosa para buscar y fue la siguiente:

- a) Se propuso una probabilidad de contacto del 90%
- b) Con goggles su visión debajo del agua fue de 2.5 m de alcance alrededor.
- a) Y su nado alcanzaba a una velocidad de 0.0000694m/seg, haciendo la conversión.

Se requiere:

Tiempo de búsqueda
Distancia recorrida
Separación de derrotas

Solución:

1er. Paso:

Por medio de tablas de la normal acumulada y $P = 90\%$ se calcula C , en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

El cálculo es de igual manera que en el problema 1 de búsqueda rectangular.

Aplicando fórmulas y consultando tablas resulta:

$$S = (1.253) \frac{W}{C} = \frac{(1.253) * 2.5}{1.64} = 1.910 \text{ m aprox.} = 2.0 \text{ m}$$

2do. Paso:

Para determinar la distancia recorrida durante la búsqueda, se procede a sumar las distancias de cada pierna recorrida, tomando en cuenta que las distancias de las dos primeras piernas que recorre el concursante son iguales al valor de S (cada pierna) y posteriormente cada dos piernas siguientes se incrementará un valor de S mas, hay que considerar que el último tramo es igual al valor de los dos tramos que le anteceden, debido a que la búsqueda termina al final del recorrido menos $S / 2$. para determinar el valor de D usamos la siguiente fórmula:

$$D = 2(1S) + 2(2S) + 2(3S) + 2(4S) + 3(5S)..... \text{ El número de vueltas de la búsqueda}$$
$$D = 4 + 8 + 12 + 16 + 30 = 70 \text{ m.} \quad \text{depende del área a cubrir y de } S.$$

3er. Paso:

Para determinar el tiempo recorrido durante la búsqueda lo determinamos de la fórmula:

$$V = \frac{D}{T} \text{ de donde } T = \frac{D}{V}$$

Así tenemos que :

$$T = \frac{70}{0.000069} = 06 :45 :53 \text{ hrs}$$

De igual forma que en el problema anterior se utilizara el sistema de aplicación.

Con el menú de la Figura 4.0.1(a) se selecciona la búsqueda de expansión cuadrada. Posteriormente se introducirán los datos requeridos para encontrar su solución.

La Figura 4.0.2(a) muestra la pantalla de búsqueda de Expansión cuadrada. La cual pide los siguientes datos:

- Ancho de barrido
- Probabilidad y Factor de Barrido:
- Latitud y Altitud.
- Velocidad descrita por el buscador, ya que para este tipo de búsqueda se utiliza solo un buscador

expansion

Anota los siguientes datos

Ancho de barrido (millas): 2.5

Area de busqueda: 12

Velocidad (Nudos): 0.0000694

Probabilidad [0 - 100] %: 90

Latitud: 12

Altitud: 1.64

Según tablas de la normal
el factor de barrido C = 1.64485
con una probabilidad del: 90 %

Si desea usar el [C] obtenido, anotelo:
en caso contrario deme su valor de C: 1.64

BUSCAR

Derrota (S): 2 Millas

Distancia Recorrida: 70 Millas Nauticas

Tiempo que Dura la Búsqueda: 06:45:53 Hrs

Simular Limpiar

Figura 4.0.2(a)

De igual forma se realiza el procedimiento del problema anterior. Es decir se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota
- Distancia que recorre el explorador
- Tiempo que dura la búsqueda.

Posteriormente se genera la Simulación gráfica, como se ilustra con la Figura 4.2(a). Donde dicha figura describe la trayectoria de búsqueda. La cual abarca toda el área, a partir del centro y extendiéndose a los extremos. Cada segmento de la espiral está etiquetada con su valor de distancia, dependiendo del valor de S (derrota).



Figura 4.0.2(b) Expansión Cuadrada

4.0.3 Búsqueda de Sector con Base Fija

Planteamiento del problema

En una Universidad de México estaban probando una estrategia de búsqueda. El simulacro fue dejar a la deriva una balsa con bultos de cemento, semejando personas a bordo. El propósito de la prueba es la simulación de un rescate y el tiempo que requiere.

La búsqueda se realizó por aire fijándose un sector sobre el mar, en donde se piensa podría estar la balsa desde su último reporte de avistamiento.

Para la búsqueda

- a) Se fijó un radio de 200 millas sobre un sector delimitado entre las siguientes marcaciones verdaderas (MV), MV=030 y MV=075; es decir que en el sector de 30 a 75 es en donde puede estar el objetivo.
- b) La probabilidad de contacto que se consideró fue de 77%
- c) Asumiendo que el radar tiene un factor de barrido de 20 millas
- b) Y una velocidad de 200 nudos.

Se requiere:

Número de aviones para la búsqueda (N)¹

Rumbos de todos los aviones (Rvs)²

Separación de derrotas corregidas (Sc)³

Factor de barrido corregido (Cc)⁴

Nueva probabilidad de contacto P(.Cc)⁵

Solución:

1er. Paso:

Trazar un arco con origen en la base (centro de la hoja) cuyo radio sea igual a la distancia de búsqueda en este caso 200 M.N.

¹ N: número de aviones

² Rvs: Rumbos verdaderos, es el rumbo que debe seguir el avión en donde se presume está el objetivo.

³ Sc: Derrota corregida; en el problema se verá que la derrota S, tendrá que calcularse solo en número entero, por lo que la derrota S inicial tendrá que aproximarse, así es que se tendrá una nueva derrota corregida llamada Sc.

⁴ Cc: Factor corregido; de la ecuación (24) del Capítulo III, el factor de barrido, se relaciona directamente con la derrota y si la derrota se corrige, el factor de barrido también se corrige.

⁵ P(c): Probabilidad de contacto: El factor de barrido depende de la probabilidad asignada según (ecuación (23) del Capítulo III) ley del cubo, y si cambia el factor de barrido, la probabilidad cambiará, a una nueva probabilidad de contacto.

2do. Paso:

Los límites de la dirección probable de aproximación al sensor se marcan entre las MV del arco; en este caso 030° y 075°.

3er. Paso:

La probabilidad de contacto es de 77 % en extremo como min. , además cuenta con un ancho de barrido (W) de 20 M.N.; con una velocidad de 200 nudos.

4to. Paso:

Por medio de tablas de la normal acumulada y $P = 77\%$ se calculara C y S, en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

$$S = (1.253) \frac{W}{C} = \frac{25.06}{1.2} = 20.8$$

5to. Paso:

La longitud del arco se obtiene con la fórmula:

$$\text{Long.arc.} = \frac{\text{arco en grados por (6.28) por radio de búsqueda}}{360^\circ}$$

$$\text{de } 030^\circ \text{ a } 075^\circ = \frac{45^\circ (6.28) (200)}{360^\circ} = 157 \text{ M.N.}$$

6to. Paso:

El número de derrotas se obtiene:

$$\text{Nº de derrotas} = \frac{\text{long.arc.}}{S} = \frac{157}{20.8} = 7.54 = 8$$

Nota: El número de derrotas debe ser siempre entero y par, por lo que se adopta el número próximo superior debido a que se requiere el 77% como mínimo, es necesario corregir “S” por lo que el nuevo valor será:

$$S_c = \frac{\text{long.arc.}}{\text{Nº derrotas}} = \frac{157}{8} = 19.625 \text{ M.N.}$$

Como C esta relacionado con S, es necesario calcular el nuevo factor de barrido, y con esto una nueva probabilidad de contacto, de tal forma que: Obsérvese ecuación (24) del Capítulo III.

$$Cc = (1.253) \frac{W}{Sc} = \frac{25.06}{19.625} = 1.27$$

Al corregir facto de barrido se debe obtener una nueva probabilidad mediante la ley del cubo en la ecuación (23) del capítulo III, resultando una nueva probabilidad de contacto del 70%.

7mo. Paso:

El número de exploradores se calcula

$$N = \frac{N^{\circ} \text{ derrotas}}{2} = \frac{8}{2} = 4$$

Lo anterior es debido a que se consideran dos derrotas por explorador (no se toma en cuenta la pierna transversal).

9vo. Paso:

El ángulo del subsector se obtiene

$$\text{Subsector de grados} = \frac{\text{arco en grados}}{N} = \frac{45^{\circ}}{4} = 11.25^{\circ}$$

10mo. Paso:

Para calcular (S / 2) en grados se determina con la fórmula:

$$(S / 2)^{\circ} = \frac{(S / 2) \text{ millas (A subsector)}}{\text{long. Subsector (millas)}} = \frac{19.625 / 2 (11.25)}{39.25} = 2.81^{\circ}$$

Pudiéndose abreviar con la fórmula:

$$(S / 2)^{\circ} = \frac{\text{subsector en grados}}{4} = \frac{11.25}{4} = 2.81^{\circ}$$

11vo. Paso:

Para cada explorador se deben determinar los tres rumbos necesarios para efectuar la búsqueda mediante las siguientes formulas:

- rumbo salida = limite izq. Del subsector + (S / 2)°
- rumbo regreso = reciproco lim.der.subsector - (S / 2)°
- el rumbo transversal se calcula con la fórmula (ejemplo del primer explorador).

$$R.\text{transversal} = R. \text{ salida} + 90^{\circ} + \text{reciproco } R.$$

$$\frac{\text{Regreso} - R. \text{ Salida}}{2} = 32.81 + 90 + \frac{(218.43 - 180) - 32.81}{2} = 32.81 + 90 + 2.25 = 125.62$$

Esta fórmula se puede abreviar con:

$$R.\text{transversal} = R.\text{salida} + 090^\circ + (S / 2)^\circ$$

Calculando Rumbo de los Exploradores

Primer explorador

$$R.\text{SAL.} = 30 + 2.81 = 32.81$$

$$R.\text{REG.} = 221.25 - 2.81 = 218.43$$

$$R.\text{TRANS.} = 32.81 + 90 + 2.81 = 125.625$$

Segundo explorador

$$R.\text{SAL.} = 41.25 + 2.81 = 44.06$$

$$R.\text{REG.} = 232.5 - 2.81 = 229.69$$

$$R.\text{TRANS.} = 44.06 + 90 + 2.81 = 136.87$$

Tercer explorador

$$R.\text{SAL.} = 52.5 + 2.81 = 55.31$$

$$R.\text{REG.} = 243.75 - 2.81 = 240.93$$

$$R.\text{TRANS.} = 55.31 + 90 + 2.81 = 148.125$$

Cuarto explorador

$$R.\text{SAL.} = 57 + 2.81 = 66.56$$

$$R.\text{REG.} = 255 - 2.81 = 252.18$$

$$R.\text{TRANS.} = 66.56 + 90 + 2.81 = 159.37$$

De igual forma que en el problema anterior se utilizará el sistema de aplicación

Con el menú de la figura 4.0.1(a) se selecciona la búsqueda de sector base fija. Posteriormente se introducirán los datos requeridos para encontrar su solución.

La figura 4.0.3(a) muestra la pantalla de búsqueda de sector de base fija. La cual pide los siguientes datos:

- Radio de búsqueda: debido a que aquí la búsqueda es por sectores
- Sector que delimitará la búsqueda
- Ancho de barrido
- Probabilidad y Factor de Barrido: estos datos van de la mano para establecer un factor de barrido X, dada la probabilidad Y.
- Velocidad descrita por los buscadores que se lleguen a necesitar según el sector.

sector_base_fija

Anota los siguientes datos

Radio de búsqueda (millas) Sector delimitado entre y

Probabilidad [0 - 100] % Ancho de barrido (Millas) Velocidad (Nudos)

Según tablas de la normal el factor de barrido C= Si desea usar el [C] obtenido anotele, en caso contrario déme su valor de C:

Resultados Teóricos

Derrota (S) Corregida (Sc) Long. del arco N° Derrotas N° de exploradores

Long. del subsector Angulo del subsector grados (S / 2)^º Nva. Prob. (%) (Cc) Corregido

Rumbo de los Exploradores

Explorador N°	R Salida	R Regreso	R Transversal
1	32.8125	218.438	125.625
2	44.0625	229.688	136.875
3	55.3125	240.938	148.125
4	66.5625	252.188	159.375

Figura 4.0.3(a) Sector Base Fija

De igual forma se realiza el procedimiento del problema anterior. Es decir se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota
- Derrota corregida, debido a que para este tipo de búsqueda el número de derrota tendrá características particulares como por ejemplo debe ser entero, lo cual modificará algunos datos.
- Longitud de arco
- Número de derrotas, debido a que estará la búsqueda dividida por sectores, estos tendrán una cantidad de derrotas (trayectorias) establecidas.
- Número de exploradores
- Longitud de subsector.
- Angulo de subsector.
- S/2, para cálculos específicos y dato visible para algún otro cálculo externo.
- Nueva probabilidad, debido a la derrota corregida.
- Cc, un nuevo factor de barrido, debido al cambio en derrota.

- Rumbo de los exploradores: cada explorador, describirá un rumbo de búsqueda, descrito por rumbo de salida, de regreso y transversal.

Posteriormente se genera la simulación gráfica, como se ilustra con la figura 4.0.3(b); donde se describe la trayectoria que sigue este tipo de búsqueda, la cual comprende sectores y cada sector es cubierto por un explorador, también se logra distinguir la separación que hay entre derrotas.

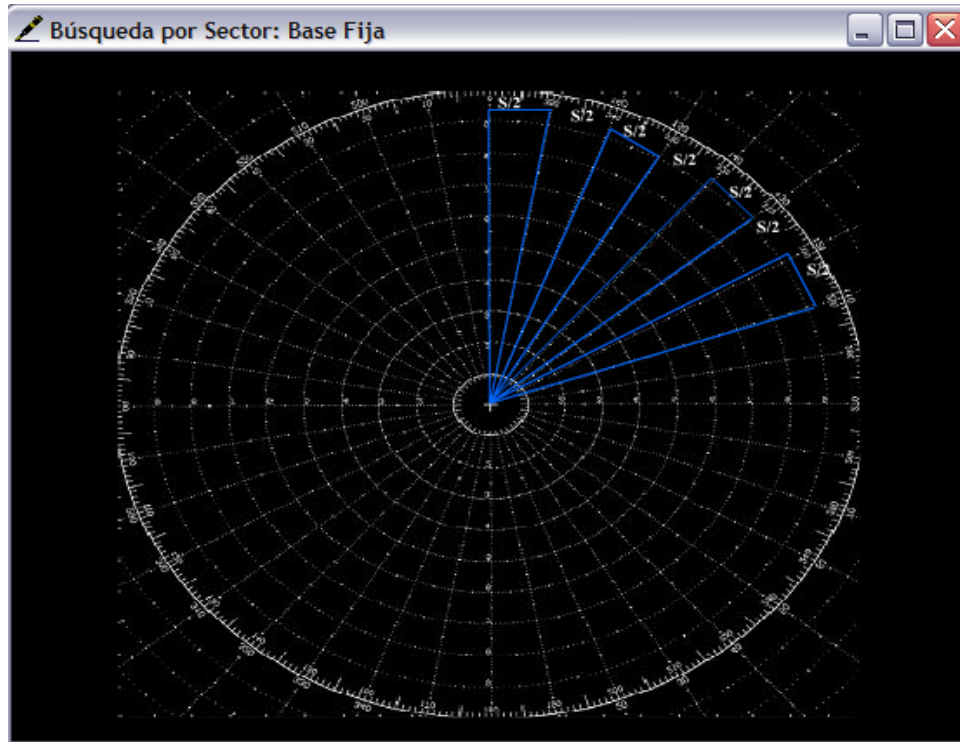


Figura 4.0.3(b) Sector: base Fija

Los siguientes problemas de búsqueda por sector en base móvil, búsqueda de intercepción por detrás, por delante y por un flanco son aplicaciones navales.

4.0.4 Búsqueda de Sector con Base Móvil

Planteamiento del problema

Con el fin de establecer una alarma temprana sobre posibles blancos que pretenda acercarse, al portaviones, se ordena efectuar una búsqueda con un radio de 100 MN, abarcando un sector de 60° , siendo este bisectado por el rumbo del portaviones. Se desea una probabilidad de contacto de por lo menos 70%, la velocidad de los exploradores es de 110 kts., el ancho de barrido por cálculo es de 20 MN. La velocidad del portaviones es de 20 kts., y navega al rumbo 45 grados.

Para efectuar la búsqueda del sensor:

- a) Se establece un radio de 100 MN, abarcando un sector de 60°

- b) Fijándose una probabilidad de contacto de por lo menos 70%
- c) La velocidad de los aviones es de 110 kts.
- d) El ancho de barrido por cálculo es de 20 MN.
- e) La velocidad del porta sensor es de 20 kts., y se mueve al rumbo 045 grados.

El primer paso del procedimiento de este tipo de búsqueda es recabar la información siguiente:

- 1.- Los límites del sector donde se efectuará la búsqueda
- 2.- El radio de búsqueda (R)
- 3.- El ancho de barrido (W)
- 4.- La probabilidad de contacto (PC)
- 5.- La velocidad de exploración (V)
- 6.- Velocidad del porta sensor (V PORT)
- 7.- Rumbo del porta sensor

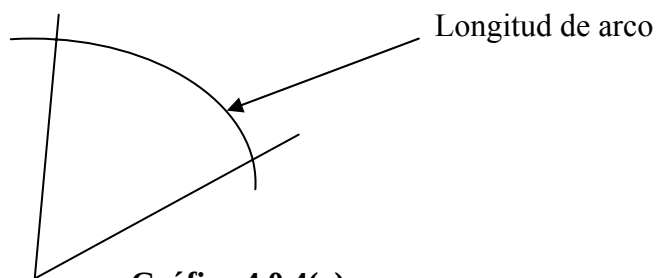
Se requiere:

Numero de exploradores
 Separación de derrotas
 Rumbo de los exploradores

Solución:

1er. Paso:

La búsqueda por sector como se muestra en la Gráfica 4.0.4(a) está limitada como su nombre lo dice por sector.



Gráfica 4.0.4(a)

Por lo que el cálculo de la longitud del arco se realiza mediante la siguiente fórmula:

$$\text{Long. Arco} = \frac{\text{Ang. Sect.} \times 2\pi \times R}{360 \text{ grados}} = \text{Millas Nauticas.}$$

Sustituyendo con los datos del ejemplo:

$$\text{Long. Arco} = \frac{(60^\circ)(2)(\pi)(100)}{360 \text{ grados}} = 104.72$$

2do. Paso:

El número de las derrotas que es necesario efectuar

$$\text{N}^\circ \text{ derrotas} = \frac{\text{Long. Arco}}{S}$$

Siendo S la derrota entre exploradores (aviones de juguete), (puede ser separación de derrotas del mismo explorador).

Por medio de tablas de la normal acumulada y $P = 70\%$ se calculara C y S, en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

$$S = (1.253)(W / C)$$

Donde C es el factor de barrido con una PC del 70% se obtiene el valor de $C = 1$.

$$S = \frac{(1.253)20 \text{ MN}}{1} = 25.06 \text{ MN}$$

$$\text{Así N}^\circ \text{ de derrotas} = \frac{\text{long. Arco}}{S} = \frac{104.72 \text{ MN}}{25.06 \text{ MN}} = 4.17$$

El número de derrotas obtenido se debe aproximar al número mas alto (par). Para mantener la probabilidad de 70% como mínimo; en caso contrario o sea con la probabilidad de contacto como máximo se aproxima al número par menor, mas próximo.

Invariablemente el número de derrotas debe ser par, ya que cada explorador efectuará dos piernas de exploración (de ida y de regreso), teóricamente; en la realidad realiza una pequeña pierna transversal como se vara en el ejemplo. Entonces:

$$\text{N}^\circ \text{ de derrotas} = 4 \text{ así } N = \frac{\text{N}^\circ \text{ Derrotas}}{2} \quad N = \frac{4}{2} = 2$$

Donde N = Número de exploradores

Después la separación de derrotas corregidas Sc se obtiene de:

$$Sc = \frac{\text{Long. Arco}}{\text{N}^\circ \text{ derrotas}} \quad Sc = \frac{104.72 \text{ M.N.}}{4} \quad Sc = 26.18 \text{ M.N.}$$

3er. Paso:

Se obtiene el ángulo del subsector sobre el que cada explorador (aviones) realizará su búsqueda y esta dado por la fórmula:

$$\text{Ang. Subsector.} = \frac{\text{Ang. Sect.}}{N} = \frac{60 \text{ grados}}{2} = 30 \text{ grados}$$

4to. Paso:

Para obtener la longitud del arco del subsector se divide la longitud total del sector entre el número de exploradores:

$$\text{Long. Subsector} = \frac{\text{long. Arco}}{N} = \frac{104.72 \text{ M.N.}}{2} = 52.36 \text{ M.N.}$$

5to. Paso:

Cálculo de Distancias:

Primera Pierna

$$R - \frac{Sc}{2} = 100 \text{ (M.N.)} - \frac{26.18 \text{ (M..N.)}}{2} = 86.91 \text{ M.N}$$

Segunda Pierna

$$\text{Dist. Trans} = Sc = 26.18 \text{ M..N.}$$

Tercera Pierna

Esta pierna será diferente para cada explorador debido a su posición al termino de la pierna transversal con respecto a la posición del porta sensor en ese momento.

Por lo anterior, es necesario obtener el tiempo al final de la pierna transversal, que sea el mismo para los tres aviones:

Distancia recorrida al final de la pierna transversal

$$(R - \frac{Sc}{2}) + Sc = 86.91 \text{ M.N.} + 26.18 \text{ M.N.} = 113.09 \text{ M.N}$$

$$T. = \frac{D}{V} = \frac{113.09 \text{ M. N.}}{110 \text{ KTS.}} = 1.02 \text{ Hrs.}$$

Cuando los exploradores terminen su segunda pierna, el porta sensor habrá recorrido una distancia de:

$$\text{Dist. Nav.} = V \times T = (20 \text{ KTS}) (1.02 \text{ Hrs.}) = 20.4 \text{ M. N.}$$

Por lo tanto, su distancia relativa al porta sensor se obtiene gráficamente.

Para obtener los rumbos de salida y transversal de cada avión se hace como si fuera un problema de sector de base fija y con la resolución de un problema cinemático para cada explorador, encontramos el rumbo y tiempo de intercepción al porta sensor (pierna de regreso).

Explorador I

$$\text{Para obtener } (S)^\circ / 2 = \frac{\text{Ang. Sub. Sect.}}{4} = \frac{30}{4} = 7.5 \text{ grados}$$

a) Rumbo de salida (Rs)

$$(Rs) = \text{Lim. izq.} + (S/2)^\circ = 15^\circ + 7.5^\circ = 22.5^\circ$$

b) Rumbo Transversal (Rtrans)

$$R_{\text{trans}} = R_s + 90^\circ + (S/2)^\circ = 22.5 + 90 + 7.5 = 120^\circ$$

c) Rumbo de Regreso (Rreg)

Este se obtiene mediante la resolución del problema cinemática correspondiente, usando la dirección y magnitud del movimiento relativo obtenido gráficamente.

Explorador II

a) Rumbo de salida (Rs)

$$(Rs) = \text{Lim. derecho.} - (S/2)^\circ = 75^\circ - 7.5^\circ = 67.5^\circ$$

Nota: Observe que cuando el avión explora a la derecha del portaviones el Rs se obtiene restando $(S/2)$ al límite derecho del subsector respectivo.

b) Rumbo Transversal (Rtrans)

$$R_{\text{trans}} = R_s - 90^\circ - (S/2)^\circ = 67.5 - 90 - 7.5 = -30^\circ + 360^\circ = 330^\circ$$

Nota: Observe la diferencia en la fórmula a usar cuando la exploración es a estribor del porta sensor, y el resultado da un número negativo.

c) Rumbo de Regreso (Rreg)

Este se obtiene mediante la resolución del problema cinemática correspondiente, usando la dirección y magnitud del movimiento relativo obtenido gráficamente.

Para obtener la resolución del problema cinemático de cada explorador es necesario obtener la siguiente información de una carta en donde se dibuja una circunferencia en grados para efectuar mediciones gráficas y obtener así velocidades y distancias.

Explorador I

MMR = 73 M.N. DMR = 206° R.Reg = 202° al rumbo de intercepción

El tiempo de intersección es $T. \text{ int.} = \frac{\text{MMR}}{\text{VMR}}$

El tiempo de exploración será:

$T. \text{ total} = \text{Al final Pierna Transversal} + T. \text{ int.} = (0.988 \text{ Hrs}) + (0.565 \text{ Hrs}) = 1.553$

Explorador II

MMR = 73 M.N. DMR = 243.5° R.Reg = 247° al rumbo de intercepción

El tiempo de intersección es $T. \text{ int.} = \frac{\text{MMR}}{\text{VMR}} = \frac{73 \text{ M.N.}}{129.2} = 0.565$

El tiempo de exploración será:

$T. \text{ total} = \text{Al final Pierna Transversal} + T. \text{ int.} = (0.988 \text{ Hrs}) + (0.565 \text{ Hrs}) = 1.553$

Los exploradores I y II arribaran al porta sensor al mismo tiempo.

MMR = Millas de Movimiento Relativo

VMR = Velocidad de Movimiento Relativo

DMR = Dirección del Movimiento Relativo

Nota: Si el porta sensor llega a cambiar de rumbo o velocidad, los exploradores resolverán su problema cinemático para encontrar el rumbo de regreso considerando el nuevo rumbo y/o velocidad.

Con el menú de la figura 4.0.1(a) se selecciona la búsqueda de sector base móvil. Posteriormente se introducirán los datos requeridos para encontrar su solución.

De igual forma que en el problema anterior se utilizara el sistema de aplicación.

La figura 4.0.4(a) muestra la pantalla de búsqueda de sector de base móvil. La cual pide los siguientes datos:

- Radio de búsqueda: debido a que aquí la búsqueda es por sectores
- Velocidad del portaviones, ya que es búsqueda en base móvil.
- Rumbo del portaviones.
- Sector que delimitara la búsqueda
- Ancho de barrido
- Probabilidad y Factor de Barrido
- Velocidad de los exploradores que harán la búsqueda
- Latitud y Altitud
- Velocidad descrita por los buscadores que se lleguen a necesitar según el sector.

sector_base_movil

Anota los siguientes datos

Radio de búsqueda 100 millas	Vel. del portaviones 20 Kts	Rumbo del portaviones 45 grados	Sector de búsqueda 60 grados
Probabilidad [0 - 100] % 70	Ancho de barrido (MN) 20	Velocidad de exploradores (Kts) 110	

Según tablas de la normal
el factor de barrido C = 1.03643
con una probabilidad del: 70 %

Si desea usar el [C] obtenido, anótelos: en caso contrario, déme su valor de C: 1

Resultados Teóricos

Derrota (S) 25.06	Long. de arco 104.72 MN	Nº Derrotas 4	Nº de exploradores 2	Long. del subsector 52.3599 MN	Angulo del subsector 30 grados
----------------------	----------------------------	------------------	-------------------------	-----------------------------------	-----------------------------------

prob. de contacto mínimo

Calculo de Distancias (M.N.)

86.91	26.1799
1era.Pierna	2da.Pierna

Datos para calcular 3era.Pierna

113.09	1.02809
Dist. Recorrida al final de pierna transversal	Tiempo

Al termino de segunda pierna el portaviones habrá recorrido una distancia de 20.5618

Rumbo de los Exploradores (El R.regreso se obtiene graficamente)

Explorador Nº	R.Salida	R.Transversal
1	22.5	120
2	67.5	330

El Rumbo de Regreso se determina graficamente

Derrota corregida (Sc)	Factor de Barrido corregido (Cc)
26.1799	0.763944

Figura 4.0.4(a) Sector Base Móvil

De igual forma se realiza el procedimiento del problema anterior. Es decir se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota
- Derrota corregida, debido a que para este tipo de búsqueda el número de derrota tendrá características particulares como por ejemplo debe ser entero, lo cual modificara algunos datos.
- Longitud de arco,
- Número de derrotas, debido a que estará la búsqueda dividida por sectores, estos tendrán una cantidad de derrotas (trayectorias) establecidas.
- Número de exploradores
- Longitud de subsector.
- Angulo de subsector.
- Cálculo de distancias para primera, segunda y tercera pierna.
- Distancia que habrá recorrido el portaviones mientras los exploradores realizan la búsqueda
- Rumbo de los exploradores: cada explorador, describirá un rumbo de búsqueda, descrito por rumbo de salida, transversal, para este tipo de búsqueda el rumbo de regreso, se obtiene gráficamente.

Posteriormente se genera la Simulación gráfica, como se ilustra con la figura 4.0.4(b); donde se describe la trayectoria que sigue este tipo de búsqueda, la cual comprende de sectores y cada sector es cubierto por un explorador, solo que en este caso hay un portaviones que sigue un rumbo y los exploradores a su regreso lo deben de alcanzar.

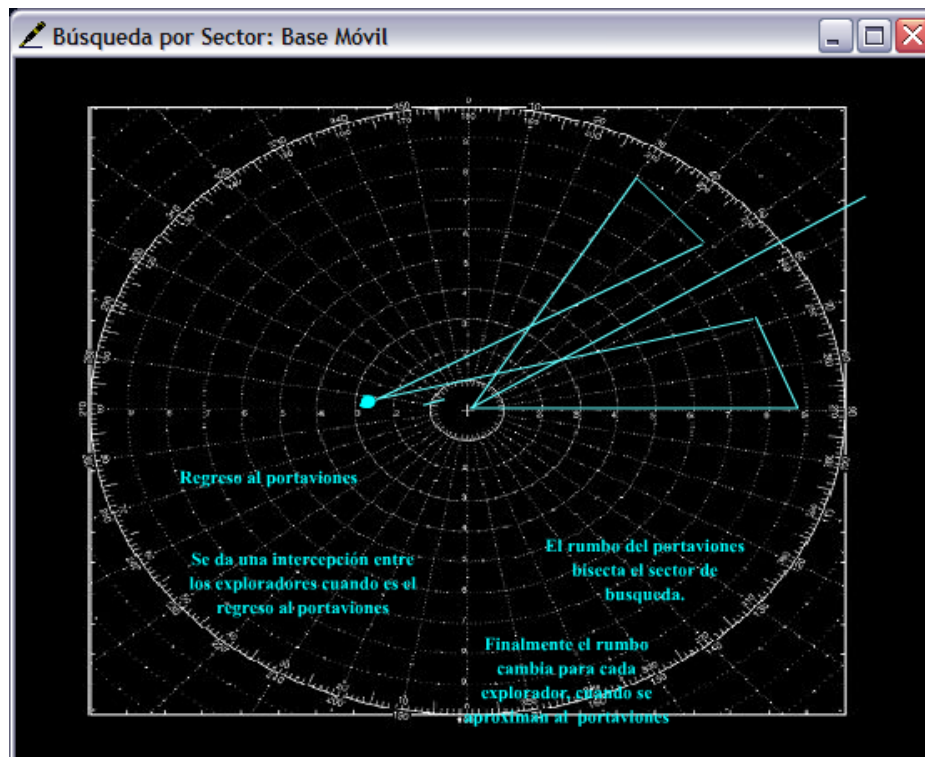


Figura 4.0.4(b) Sector: base móvil

4.0.5 Búsqueda por Delante

Se emplea normalmente por buques de superficie y esta destinada por delante al blanco cuya posición previa, rumbo y velocidad son conocidos dentro de los límites razonables de exactitud.

Descripción

- 1.- En este tipo de búsqueda la línea de exploración se forma inicialmente sobre una línea en ángulo recto a la derrota más probable del blanco y con su centro sobre esta derrota y en el punto más avanzado del área probable del blanco.
- 2.- Es una forma especial de búsqueda de área usada cuando es posible reducir el área a ser cubierta.
- 3.- El área probable del blanco es el área en la cual, es probable que este el blanco a una hora dada; su tamaño y forma dependerá de la exactitud de la información original respecto al blanco, y al tiempo transcurrido desde que se obtuvo aquella información.
- 4.- Alternativa: La finalidad de una búsqueda de intercepción por delante se puede cumplir por medio de una patrulla de barrera.

Planteamiento del problema

A las 8:30 hrs., se reporta un blanco en $Mv = 240^\circ$ a una distancia de 40 M., con un error en la posición estimada de 5M, por lo que se ordena interceptarlo, dada la posición relativa del blanco se decide efectuar una intercepción por delante, nuestra unidad táctica está en disposición de zarpar del fondeadero a las 9:00 hrs. A una velocidad de exploración de 30 nudos, se estima que el rumbo probable del blanco es $R = 000^\circ$ y que podría variarlo en $\pm 30^\circ$, su velocidad mínima es de 15 nudos, su velocidad probable es de 18 nudos y su máxima velocidad es de 20 nudos, se desea tener una probabilidad de contacto del 90% como mínimo, el ancho de barrido es de 30M. Se requiere:

1. Hora de intercepción
2. Número de exploradores
3. Separación de derrotas corregidas
4. Nueva probabilidad de contacto.

Datos:

Buque propio:

Velocidad = 30 N

Factor de Barrido $C = 1.1$

Ancho de Barrido $W = 30$ M

Prob. De contacto = 90% (mínimo)

Blanco:

Movimiento Verdadero M_v : 240° ⁶

Distancia $D = 40 \text{ M}$ ⁷

Error de posición $E_p = 5 \text{ M}$ ⁸

$V_{\min} = 15 \text{ N}$

$V_{\text{prob}} = 18 \text{ N}$ Velocidades estimadas del objetivo

$V_{\max} = 20 \text{ N}$

Rumbo probable $R_{\text{prob}} = 000^\circ \pm 30^\circ$ ⁹

Tiempo muerto $t_m = 0.5 \text{ hrs}$ ¹⁰

Solución:

1er. Paso:

Primero se limita el área a ser buscada; que serán los límites del rumbo probable, proyectados tangente al círculo formado por el radio del error en la posición dada (E_p) cuyo centro es “G”. Ver Gráfica 4.0.5(a).

“G1” se encuentra ubicado en la posición de avance donde corta el rumbo probable del blanco al círculo de error la posición; “G2” está ubicado en el extremo opuesto a la posición “G1”. Ver Gráfica 4.0.5(a).

2do. Paso:

Se determina el tiempo muerto (t_m) que es la diferencia que existe entre la hora que se recibe el reporte del avistamiento del blanco y la hora en que se está listo para zarpar.

3er. Paso:

Se determina el avance del blanco durante el tiempo muerto para definir el punto de intercepción (A^1) por medio de $D = V \times T$, considerando la velocidad máxima del blanco, y sobre el rumbo probable. (con centro en G1). Ver Gráfica 4.0.5(a).

⁶ M_v : movimiento verdadero, es el movimiento que sigue el blanco.

⁷ Distancia: distancia a la que se vio el objetivo

⁸ E_p : Error de posición, cierta distancia de holgura según la distancia desde donde se vio el objetivo

⁹ R_{prob} : Rumbo probable, es el rumbo que presuntamente sigue el objetivo.

¹⁰ Tiempo muerto (t_m) que es la diferencia que existe entre la hora que se recibe el reporte del avistamiento del blanco y la hora en que se está listo para zarpar

$$D = V_{\max} \times t_m$$

$$D = 20 \text{ N} \times 0.5 \text{ hrs.}$$

$$D = 10 \text{ Nudos.}$$

4to. Paso:

Se resuelve el problema cinemático de intercepción considerando el avance del blanco, de donde se obtiene el tiempo de intercepción (t_i) por medio de la formula : $t_i = d / v$. (usando el rumbo probable y la velocidad máxima).

$$t_i = D / V$$

$$t_i = 34.8 \text{ M} / 25.2 \text{ N}$$

$$t_i = 1.38 \text{ hrs}$$

5to. Paso.

Se determina el tiempo total (T_t) que es igual a la suma del tiempo muerto mas el tiempo de intercepción.

$$T_t = t_m + t_i = 0.5 \text{ hrs.} + 1.38 \text{ hrs.} = 1.88 \text{ hrs.}$$

6to. Paso:

Se determina el avance máximo del blanco, calculando el área probable del blanco la cual esta delimitada por los arcos AB y CD. Ver Gráfica 4.0.5(a).

- a) El arco AB se obtiene multiplicando el tiempo total por la velocidad máxima del blanco y para trazarlo deberá de tomarse como centro “G1” hasta que corte a los límites del rumbo probable del blanco, siendo este arco la línea de máximo avance del blanco. Ver Gráfica 4.0.5(a)

$$\text{Arc. AB} = T_t \times V_{\max} = 1.88 \text{ hrs.} \times 20 \text{ N} = 37.6 \text{ Millas}$$

- b) El arco CD se obtiene multiplicando el tiempo total por la velocidad mínima del blanco y para trazarlo debe tomarse como centro “G2” hasta que corte a los límites del rumbo probable del blanco, siendo este arco la línea de mínimo avance del blanco. Ver Gráfica 4.0.5(a).

$$\text{Arc. CD} = T_t \times V_{\min} = 1.88 \text{ hrs.} \times 15 \text{ N} = 28.2 \text{ Millas}$$

7mo. Paso:

Para determinar el frente de búsqueda (F) deberá trazarse una tangente a la línea de máximo avance del blanco y perpendicular a su rumbo probable hasta cortar los límites de dicho rumbo. La distancia de este frente se determina gráficamente.

$$F = 60.5 \text{ Millas.}$$

8vo. Paso:

Se determinara la separación de derrotas (S)

Nota: En el ámbito naval esta C, (factor de barrido) puede cambiar debido a que se manejan variables como el estado de la mar e incluso la experiencia propia del encargado de la toma de decisiones. Por lo que la C que aquí se considera fue tomada de una curva de buen tiempo de la mar.

Por medio de tablas de la normal acumulada y $P = 90\%$ se calculará C y S, en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

El Factor de Barrido para una probabilidad de 90% según tablas de la normal sería : $C = 1.64$

En este caso, al consultar las curva de buen tiempo de la mar es $C = 1.1$.

Dependiendo del numero de olas, mar del 0 no hay olas hasta mar del 5.de tamaño es oleaje

$$S = (1.253)(W / C) = (1.253) 30 M / 1.1 = 34.17 \text{ Millas}$$

9no. Paso:

Se determina el número de exploradores (N) . Despejando la fórmula

$$F = (N - 1) (S) + W \dots\dots\dots\text{Cunado W es diferente de S}$$

$$F = N \times S \dots\dots\dots\text{Cuando W es igual a S}$$

Cuando “ N “ es fraccionario se deberá ajustar al número entero más próximo, considerando la probabilidad de contacto establecida.

En este problema en particular se usa:

$$F = (N - 1) (S) + W \dots\dots\dots\text{despejando} \dots\dots N = \frac{F - W}{S} + 1 = \frac{60.5 M - 30 M}{M} + 1 = 1.89 = 2$$

10mo. Paso:

Con el ajuste hecho a “N”, deberá corregirse la separación de derrotas (Sc) despejándola de la misma formula

$$F = (N - 1) (S) + W$$

$$Sc = \frac{F - W}{N - 1} = \frac{60.5 M - 30 N}{2 - 1} = 30.5 M$$

11vo. Paso:

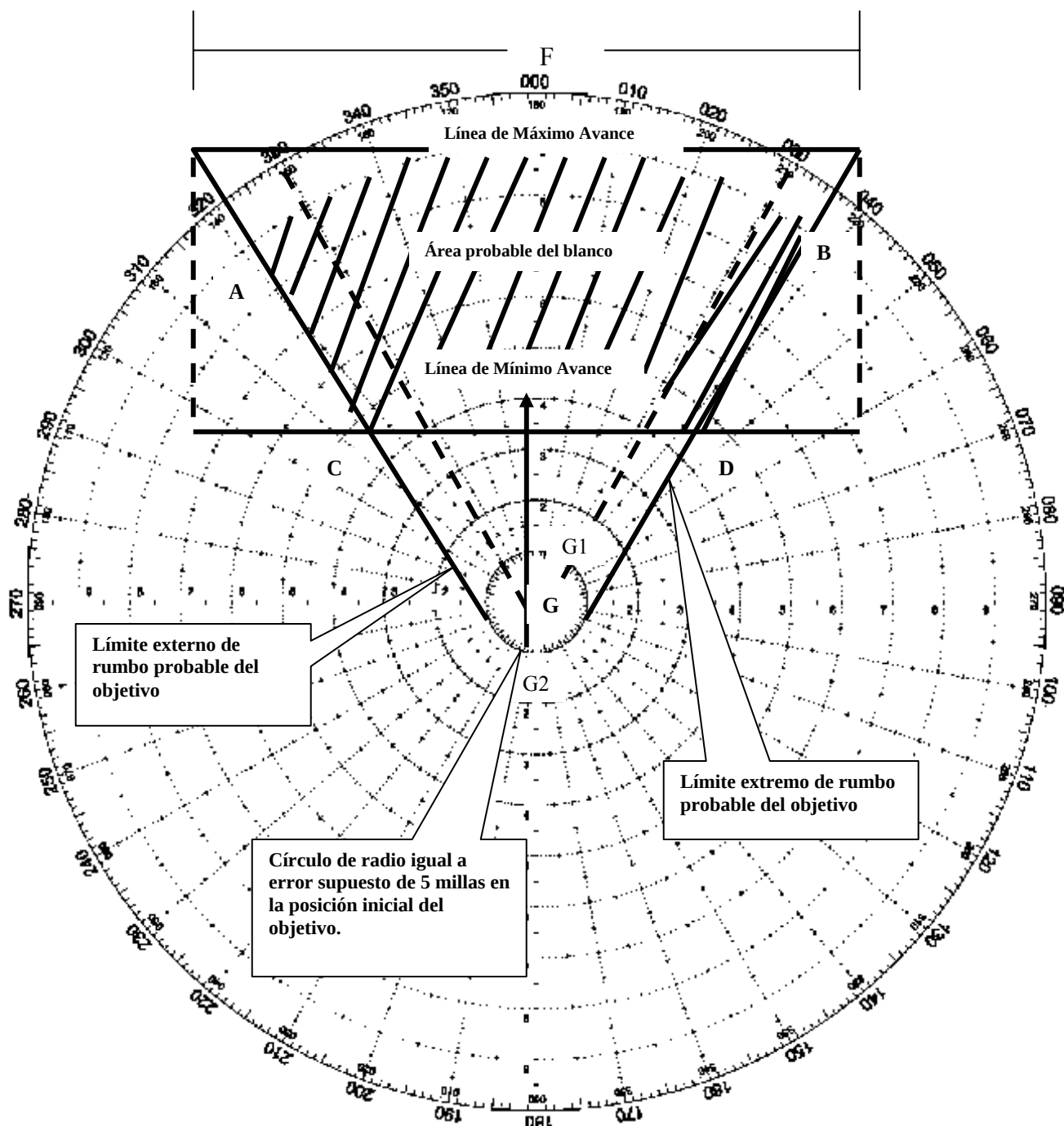
Por medio de tablas de la normal acumulada y $P = 90\%$ se calculará C_c y S_c , en conjunto con la ley inversa del cubo. Obsérvese ecuaciones (23) y (24) del Capítulo III.

Con la separación de derrotas corregida (S_c) se obtiene el factor de barrido corregido (C_c) y con este valor se obtiene la nueva probabilidad de contacto.

$$C_c = (1.253)(W / S_c) = (1.253)(30 \text{ M} / 30.5 \text{ M}) = 1.23$$

Localizando en tablas el valor de C_c la probabilidad de contacto es de 78%.

La forma gráfica de representar los resultados antes mencionados se muestran en la Gráfica 4.0.5(a).



Grafica 4.0.5(a) Búsqueda por Delante.

De igual forma que en el problema anterior se utilizará el sistema de aplicación

Con el menú de la figura 4.0.1(a) se selecciona la búsqueda por delante.

Posteriormente se introducirán los datos requeridos para encontrar su solución.

La figura 4.0.5(a) muestra la pantalla de búsqueda por delante. La cual pide los siguientes datos:

Buque Propio

- Velocidad del buque.
- Ancho de barrido
- Probabilidad y Factor de Barrido: estos datos van de la mano para establecer un factor de barrido X, dada la probabilidad Y.

Buque a localizar

- Reporte de avistamiento del buque
- Hora en que zarpa el buscador
- Distancia a la que se diviso de buque.
- Error de posición, rango de error.
- Velocidad mínima, probable y máxima que se estima.
- Frente de búsqueda

The screenshot shows a software window titled 'busqueda_delante'. It contains two main sections: 'Buque Propio' and 'Buque a localizar'. Below these is a 'BUSCAR' button, followed by a 'Resultados Teóricos' section with various output fields and 'Simular' and 'Limpiar' buttons at the bottom.

Buque Propio		Buque a localizar	
Velocidad	30 nudos	Reporte (hrs.min)	0830
Probabilidad	90 [0 - 100] %	Zarpár (hrs.min)	0900
Ancho de barrido	30 Millas	MV	240 grados
Según tablas de la normal		Distancia	40 Millas
factor de barrido Cc		Error de posición	5 Millas
con una probabilidad del 90 %		Velocidades (Nudos)	
Si desea usar el [C] obtenido anterior en caso contrario deme su valor de C:		Vmin	15
1.1		Vprob	18
[chequear tablas]		Vmax	20
		Frente de Búsqueda F =	60.5 Millas
		Tiempo de intercepción (hrs)	1.38
BUSCAR			
Resultados Teóricos			
Hora de int. (Hrs.)		Nº de exploradores	
10:22:40		2	
Derrota (S)	34.1727 Millas	Corregida (Sc)	30.5 Millas
Nva. Prob. Contacto	67.4691 [0 - 100] %		
Tiempo Total (Hrs)	1.88	Avance Max (Millas)	37.6
Avance al principio de intercepción		Factor Corregido (Cc)	
10 Millas		0.963507 Millas	
[Simular]		[Limpiar]	

Figura 4.0.5(a) Búsqueda por Delante

De igual forma se realiza el procedimiento del problema anterior. Es decir, se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota

- Derrota corregida, debido a que para este tipo de búsqueda el número de derrota tendrá características particulares como por ejemplo debe ser entero, lo cual modificara algunos datos.
- Rumbo
- Tiempo muerto
- Número de exploradores
- Hora de intercepción
- Nueva probabilidad de contacto
- Avance máximo y mínimo
- Tiempo total
- Factor de barrido corregido
- Avance al principio de intercepción

Posteriormente se genera la simulación gráfica, como se ilustra con la figura 4.0.5(b); donde se describe la trayectoria que sigue este tipo de búsqueda. Teniendo intercepción con el blanco por delante.

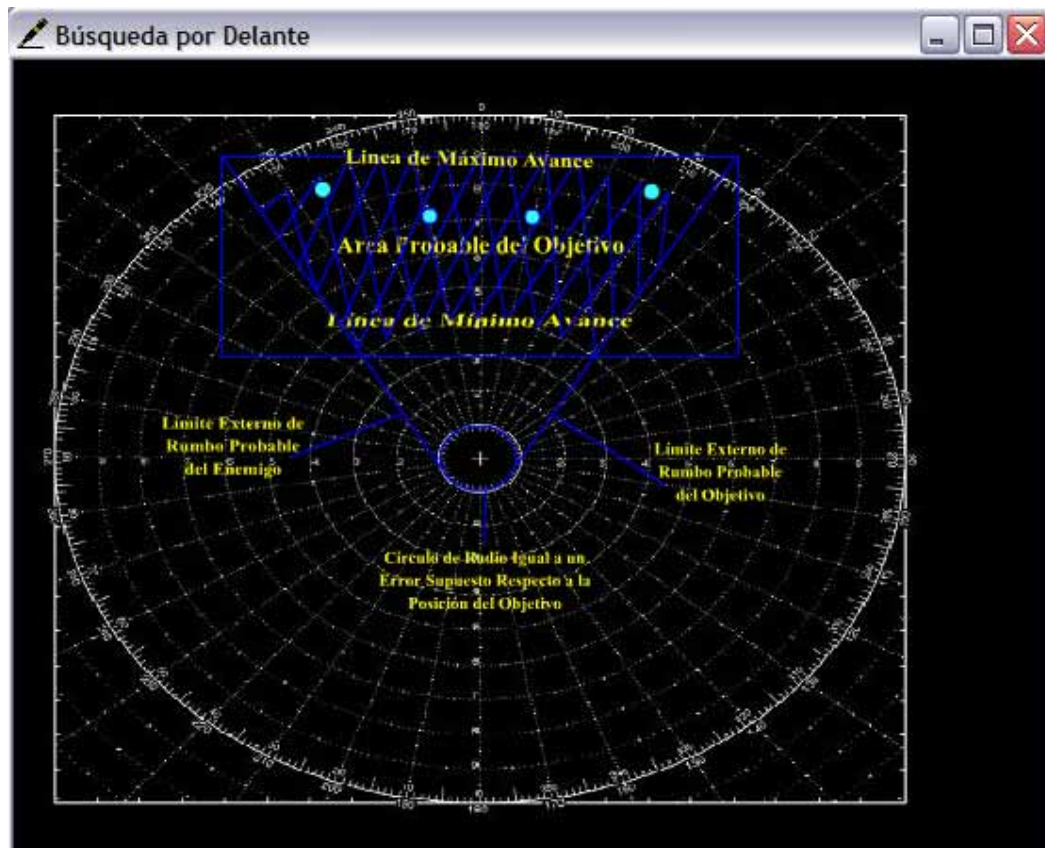


Figura 4.0.5(b) Búsqueda por Delante

4.0.6 Búsqueda por Detrás

En este tipo de intercepción la velocidad de exploradores debe ser tal que los exploradores alcancen la línea móvil de avance máximo del blanco antes de que los límites de su rumbo

probable pasen fueran de las derrotas de los exploradores de los flancos, así mismo el siguiente ejemplo se resuelve suponiendo que el blanco se intercepta en un punto de mínimo avance; pero podría estar en cualquier otro punto del área probable, por lo que habría de calcular el tiempo de exploración que dependerá de la velocidad relativa de los exploradores con respecto al blanco.

Planteamiento del problema

A las 20:00 hrs. Se reporta ver una lancha en $Mv = 309^\circ$ a una distancia de 47 MN. Con un error estimado en la posición de 10 MN.

Se estima el rumbo verdadero más probables del blanco en 000° , mismo que podrá variarlo $\pm 25^\circ$ a banda y banda, que su velocidad mínima es de 20 nudos, la probable de 22 nudos y la máxima de 25 nudos.

Nuestros buques, que están equipados con radares SPS-2 con un $W=26$ MN y desarrollan una velocidad de exploración de 30 nudos, podrán zarpar a las 21:00 hrs.

Se desea tener una probabilidad de contacto de 90% como mínimo, pues nos lo permite el estado de la mar.

Se requiere:

- 1.- Rumbo probables de intercepción
- 2.- Tiempo de intercepción
- 3.- Separación de derrotas
- 4.- Longitud del frente de exploración
- 5.- Número de exploradores
- 6.- Separación de derrotas corregido
- 7.- Nueva probabilidad de contacto

Solución:

1er. Paso:

Se determina C (factor de barrido) con una probabilidad de 90% y la curva de buen tiempo, nos da un valor de $C = 1.1$

Nota: En el ámbito naval esta C (factor de barrido) puede cambiar debido a que se manejan variables como el estado de la mar e incluso la experiencia propia del encargado de la toma de decisiones. Por lo que la C que aquí se considera fue tomada de una curva de buen tiempo de la mar.

El Factor de Barrido para una probabilidad de 90% según tablas de la normal sería : $C = 1.64$

En este caso, al consultar la curva de buen tiempo de la mar es $C = 1.1$.

2do. Paso:

Se determina el tiempo muerto, que es el tiempo que transcurre desde que nos es reportado el blanco hasta que le buque propio puede zarpar para efectuar la intercepción, en el ejemplo nos es reportado a las 20:00 hrs. Y se puede zarpar a las 21:00 hrs. así es que el tiempo muerto es de 1 hora.

3er. Paso.

Se determina el avance al principio de la intercepción llamada A'; que es igual al tiempo muerto por la velocidad mínima.

Avance al principio de la intercepción

$$A' = T_m \times V_{min} = 1 \times 20 = 20 \text{ M.N.}$$

4to. Paso:

Se determina el tiempo total que es igual, al tiempo muerto más tiempo de intercepción.

$$T_t = T_m + T_i = 1 + 4.5 = 5.5 \text{ hrs.}$$

5to. Paso:

Se determina el sector AB que es igual, al tiempo total por la velocidad máxima. Ver Grafica 4.0.6(a).

$$AB = T_t \times V_{max} = 5.5 \times 25 = 137.5 \text{ MN}$$

6to. Paso:

Se determina el sector CD, que es igual al tiempo total multiplicado por la velocidad mínima. Ver Grafica 4.0.6(a).

$$CD = T_t \times V_{min} = 5.5 \times 20 = 110 \text{ MN}$$

El área delimitada por estos dos arcos AB Y CD nos da el área probable del blanco. Ver Grafica 4.0.6(a).

7mo. Paso:

La separación de derrotas (S), es igual al ancho de barrido (W) entre el factor de barrido (C). Obsérvese ecuación (24) del Capítulo III.

$$S = (1.253) \frac{W}{C} = \frac{1.253(26)}{1.1} = 29.61 \text{ MN}$$

$$\text{Si } W \neq S; \quad F = (N - 1) S + W$$

$$\text{Si } W = S ; \quad F = NS$$

Como $S \neq W$, entonces $F = (N - 1) S + W$

$$\text{Despejando } N, \text{ se obtiene: } N = \frac{F - W}{S} + 1$$

Sustituyendo:

$$N = \frac{160 - 26}{29.61} + 1 = 5.52 \approx 6$$

8vo. Paso:

Tomamos $N = 6$ ya que nos piden una probabilidad de contacto 90% como mínimo, entre mas exploradores sean, mayor probabilidad de contacto tendremos.

Se debe corregir la separación de derrotas.

$$Sc = \frac{F - W}{N - 1} = \frac{160 - 26}{6 - 1} = 26.8 \text{ MN.}$$

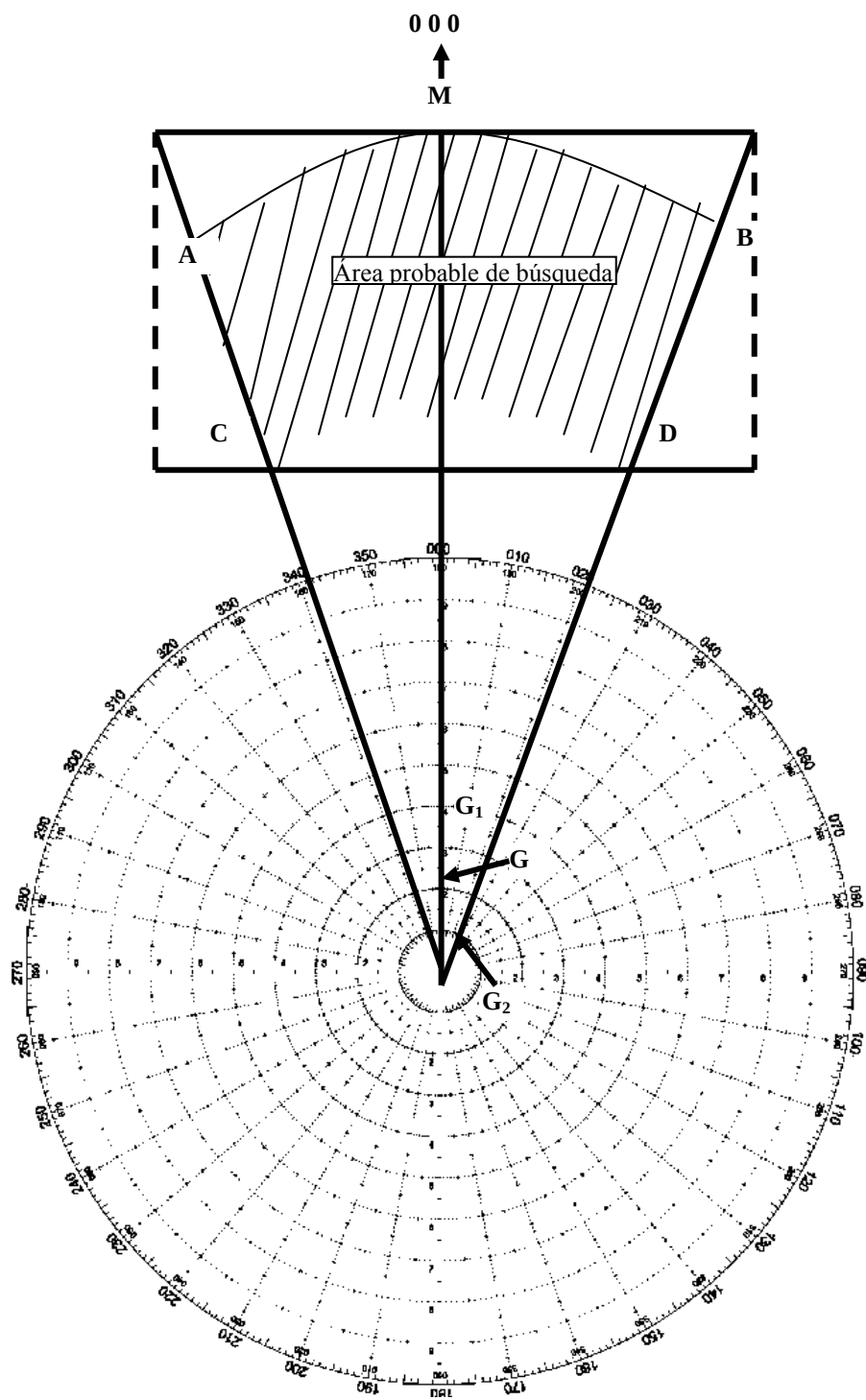
9no. Paso:

En base a la separación de derrotas corregida (Sc), se obtiene un nuevo factor de barrido con el cual se obtendrá una nueva probabilidad de contacto (PC) Obsérvese ecuaciones (23) y (24) del Capítulo III.

$$Cc = (1.253) \frac{W}{Sc}$$

$$Cc = \frac{26(1.253)}{26.8} = 1.21 \quad \text{Nuava PC} = 77\%$$

La forma grafica de representar los resultados antes mencionados se muestran en la Grafica 4.0.6(a).



Grafica 4.0.6(a) Búsqueda por Detrás.

De igual forma que en el problema anterior se utilizara el sistema de aplicación

Con el menú de la Figura 4.0.1(a) se selecciona la búsqueda por detrás.

Posteriormente se introducirán los datos requeridos para encontrar su solución.

La figura 4.0.6(b) muestra la pantalla de búsqueda de búsqueda por detrás. La cual pide los siguientes datos:

Buque Propio

- Velocidad del buque.
- Ancho de barrido
- Probabilidad y Factor de Barrido: estos datos van de la mano para establecer un factor de barrido X, dada la probabilidad Y.

Buque a localizar

- Reporte de avistamiento del buque
- Hora en que zarpa el buscador
- Distancia a la que se diviso de buque.
- Error de posición, rango de error.
- Velocidad mínima, probable y máxima que se estima.
- Longitud frente de exploración
- Tiempo de intercepción

intercepcion_detras

Anota los siguientes datos

Buque Propio		Buque a localizar		
Velocidad	Probabilidad	Reporte	Zarpar	MV
30 Nudos	90 [0 - 100] %	(Hrs,min)	(Hrs,min)	309 grados
		2000	2100	
Ancho de barrido	Según tablas de la normal	Distancia	Error de posición	
26 Millas	factor de barrido C=	47 Millas	10 Millas	
	con una probabilidad del:			
	%			
Si desea usar el [C] obtenido, anótelos en caso contrario deme su valor de C:	1.1	Velocidades (Nudos) Vmin Vprob Vmax 20 22 25		
	chechar tablas	Longitud Frente de Exploración F = 160 Millas		
		Tiempo de intercepción (hrs) 4.5		
BUSCAR				
Resultados Teóricos				
Derrota (S)		Hora de int. (Hrs.)	Nº de exploradores	
29.6164 Millas		00:30:00	6	
Corregida (Sc)		Nva. Prob. Contacto		
26.8 Millas		77.5862 (0 - 100) %		
Tiempo Total (Hrs)		Avance Max (Millas)		Avance Min (Millas)
5.5		137.5 Sector AB		110 Sector CD
Avance al principio de intercepción (A')		Factor Corregido (Cc)		
20 Millas		1.2156 Millas		
Simular Limpiar				

Figura 4.0.6 Búsqueda por Detrás

De igual forma se realiza el procedimiento del problema anterior. Es decir se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota: es la trayectoria (es un valor en distancia).
- Derrota corregida, debido a que para este tipo de búsqueda el número de derrota tendrá características particulares como por ejemplo debe ser entero, lo cual modificara algunos datos.
- Rumbo
- Número de exploradores
- Hora de intercepción
- Nueva probabilidad de contacto
- Avance máximo y mínimo
- Tiempo total

- Factor de barrido corregido
- Avance al principio de intercepción

Posteriormente se genera la Simulación grafica, como se ilustra con la Figura 4.0.6 (b).Donde dicha Figura describe la trayectoria que sigue este tipo de búsqueda, e intercepta al objetivo por detrás.

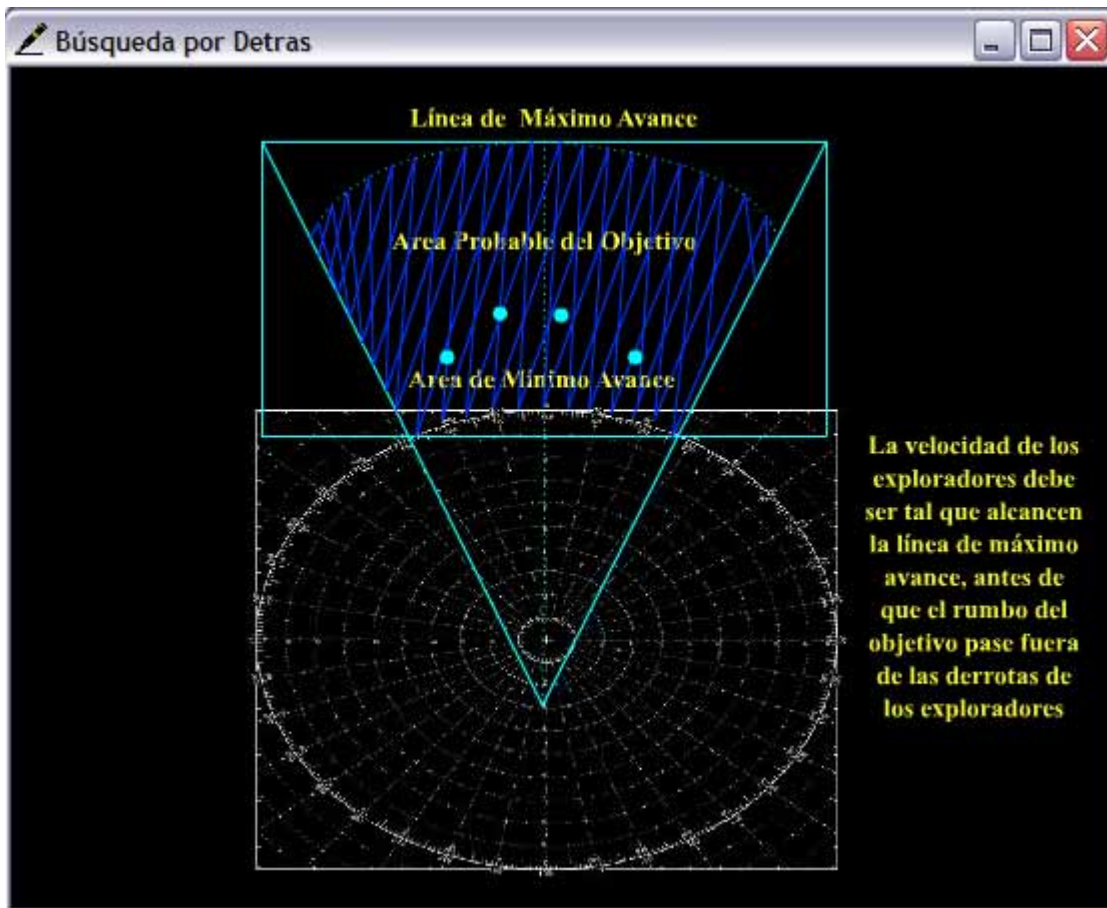


Figura 4.0.6(b) Búsqueda por Detrás

4.0.7 Búsqueda por un Flanco

Para la selección de este tipo de intercepción al igual que en intercepción por detrás, los exploradores deben tener ventaja en velocidad sobre el blanco, otro factor es la posición relativa que se tenga sobre el blanco.

Planteamiento del problema

Problema de búsqueda de intercepción por un flanco (alejándose de la posición informada del blanco).

A las 9:15 hrs nos reportan un blanco en $Mv=170^\circ$ a una distancia de 30 millas de nosotros, mismo que según información de inteligencia, se encuentra navegando a un rumbo verdadero probable de 90° , considerándose un error en el rumbo de $\pm 20^\circ$; el error en la posición del blanco es de 5 millas, las velocidades que desarrolla el blanco son vel. mínima = 15 nudos, vel. probable = 18 nudos, vel. máxima = 21 nudos, debido a que nuestros sensores se encuentra fuera de servicio, la intercepción deberá hacerse con un ancho de barrido visual de 10 millas, la probabilidad del contacto es de 80% con malas condiciones meteorológicas existentes en el área. Para realizar la búsqueda contamos con un suficiente número de buques exploradores los cuales desarrollan una velocidad de exploración de 27 nudos mismos que se encuentran fondeados y que estarán en disponibilidad de zarpar a las 10:00 hrs.

Se requiere:

- 1.- Rumbo verdadero de intercepción
- 2.- Tiempo de intercepción
- 3.- Frente de búsqueda inicial
- 4.- Número de exploradores necesarios
- 5.- Nuevo frente de búsqueda
- 6.- Separación de derrotas inicial y corregido
- 7.- Nueva probabilidad de contacto

Solución:

1er. Paso:

Tiempo muerto: es el lapso de tiempo transcurrido desde que se recibe la orden de avistamiento, hasta el momento de zarpe.

$$T_m = 10:00 \text{ hrs} - 09:15 \text{ hrs.} = 45 \text{ min.} = 0.75 \text{ hrs.}$$

Distancia al punto limite más cercano del blanco se le llamará "A". Véase Gráfica 4.0.7(a)

$$A = \text{vel. probable} \times T_m = 18 \text{ nudos} \times 0.75 \text{ hrs.} = 13.5 \text{ millas}$$

Tiempo de intercepción = 1.66 (valor que resulta, al resolver un problema cinemático, sobre la rosa de maniobras).

Tiempo Total = es la suma del tiempo muerto + tiempo de intercepción
Tiempo total = 0.75 hrs. + 1.66 hrs. = 2.41 hrs.

2do. Paso:

Planteamiento de la búsqueda

Frente inicial de exploración

Es la distancia que existe del punto “A” al punto “C” medida sobre el rumbo límite probable mas cercano del blanco. Véase Gráfica 4.0.7(a)

Obtención del frente de exploración (F).

Estos arcos representan el avance al principio de la intercepción

Avance mínimo:

Arco “AB” = vel.max x tiempo total = 21 x 2.41 = 50.61 M (haciendo centro en G-1).
Véase Gráfica 4.0.7(a)

Avance máximo:

Arco “CD” = vel. min. x tiempo total = 15 x 2.41 = 36.15 M (haciendo centro en G-2).
Véase Gráfica 4.0.7(a)

Frente de exploración (F) = (dist. “AB” + 5 M) - (dis. “CD” - 5 M)
= (50.61 + 5) - (36.15 - 5) = 55.61 - 31.15 = 24.46 Millas.

Punto de intercepción (A): es la posición en la cual el blanco deberá ser interceptado, en el punto “A” en caso de que navegue sobre el rumbo límite mas cercano y con la velocidad probable, se obtiene multiplicando la velocidad probable por el tiempo total y estará situado sobre el rumbo límite mas cercano del blanco.

Punto de intercepción (A) = vel. prob. x tiempo total = 18 nudos x 2.41 hrs. = 43.38 Millas
(haciendo centro en G). Véase Gráfica 4.0.7(a)

Si el blanco no es interceptado en el punto “A”, tendrá que efectuarse la búsqueda sobre los rumbos (A-B) (B-C) (C-D) (D-C) ..etc.. El cual nos llevara un tiempo el recorrerlos, que es el tiempo de búsqueda. Véase Gráfica 4.0.7(a)

Con el frente de búsqueda se calculó considerando que el blanco navegaba sobre el límite mas cercano (tiempo total) ; a medida que se prolonga la búsqueda hay menos probabilidades que lo interceptemos, existiendo la posibilidad de localizarlo con certeza si calculamos el frente de búsqueda entre los puntos “B” y “D” del área probable de búsqueda calculada con el tiempo total mas el tiempo de búsqueda y las velocidades max. y min. del blanco con centro en G₂ y G₁ respectivamente.

Nota: Para diferenciar llamaremos nuevo frente de exploración al requerido hasta el final de la búsqueda.

Nuevo frente de exploración: para calcular el nuevo frente de exploración se obtiene el nuevo tiempo hasta que termina la búsqueda mediante la suma del tiempo muerto + tiempo de intercepción + tiempo de búsqueda. Una vez calculado el nuevo tiempo total se obtienen las distancias “Q”¹¹ y “R”¹² y se efectúa la suma algebraica de la siguiente manera:

Nuevo tiempo total (T') = Tm + Tint. + Tbusq.

Tbusq. = 2hrs. + 0.67 hrs. = 2.67 hrs

Nota: el tiempo de búsqueda es el que tardan los exploradores en recorrer el área de búsqueda y se obtiene gráficamente (en este caso son tres cambios de rumbo completos y una fracción los cuales recorreran los buques exploradores). L

T' = 0.75 hrs. + 1.66 hrs. + 2.67 hrs. = 5.08 hrs.

Distancia “Q” = vel.min. x tiempo total = 15 x 5.08 = 76.2 millas (haciendo centro de G-2).

Distancia “R” = vel.max. x tiempo total = 21 x 5.08 = 106.7 millas (haciendo centro de G-1).

Nuevo Frente = F' = Dist “QR”.

$$\begin{aligned} F' &= (\text{Dist. R} + 5 \text{ millas}) - (\text{Dist.Q} - 5 \text{ millas}). \\ &= (106.7 + 5) - (76.2 - 5 \text{ millas}). \\ &= 111.7 - 71.2 = 40.5 \text{ millas} \end{aligned}$$

Separación de derrotas (S):

Nota: En el ámbito naval esta C (factor de barrido) puede cambiar debido a que se manejan variables como el estado de la mar e incluso la experiencia propia del encargado de la toma de decisiones. Por lo que la C que aquí se considera fue tomada de una curva de buen tiempo de la mar.

El Factor de Barrido para una probabilidad de 80% según tablas de la normal sería : C = 1.28

En este caso, al consultar la curva de buen tiempo de la mar es C = 1.8

¹¹ “Q”: representa el avance mínimo a la intercepción, cuando no se realizó la intercepción anterior, y se ha calculado el tiempo de búsqueda.

¹² “R”: representa el avance máximo a la intercepción, cuando no se realizó la intercepción anterior, y se ha calculado el tiempo de búsqueda.

Obsérvese ecuación (24) del Capítulo III.

$$S = \frac{(1.253)W}{C} = \frac{\text{ancho de barrido}}{\text{factor de barrido}}$$

El factor de barrido C con probabilidad de contacto de 80% y con malas condiciones meteorológicas es 1.8.

$$S = \frac{(1.253)W}{C} = \frac{(1.253)10}{1.8} = 6.96 \text{ millas}$$

3er. Paso:

Numero de exploradores (N)

Cuando: $S = W$ se utiliza la formula: $F = N \times S$

Cuando $S \neq W$ se utiliza la formula : $F = (N - 1) (S) + W$

Por lo tanto en este caso se utiliza . $F = (N - 1) (S) + W$

Despejando a N tenemos.

$$N = \frac{F - W}{S} + 1 = \frac{40.5 - 10}{6.96} + 1 = 5.38 \approx 6 \text{ exploradores}$$

Cuando el numero de exploradores no da en cantidad entera se debe aproximar al numero entero inmediato superior o inferior dependiendo de las condiciones de la probabilidad de contacto que requiere el problema.

4to. Paso:

Separación de derrotas corregida

Con el nuevo número de exploradores se corrige la separación de derrotas.

$$F = (N - 1) (S) + W$$

$$Sc = \frac{F - W}{N - 1} + 1 = \frac{40.5 - 10}{6 - 1} = 6.1 \text{ millas}$$

5to. Paso:

Nueva probabilidad de contacto.

Con la separación de derrotas corregidas se obtiene la nueva probabilidad de contacto. Obsérvese ecuación (24) del Capítulo III.

$$Sc = (1.253) \frac{W}{Cc} ; \quad Cc = (1.253) \frac{W}{S} = \frac{10}{5.08} = 2.46 \text{ millas Véase Gráfica 4.0.7(a)}$$

Con $C = 1.97$ se consulta en tablas y se encuentra que la nueva probabilidad de contacto es de 81%. Obsérvese ecuación (23) del Capítulo III.

6to. Paso:

Rumbos verdaderos a adoptar durante la búsqueda en caso de no interceptarlo
Véase Gráfica 4.0.7(a), en el punto “A”.

R1 .- 125.5°

R2 .- 146°

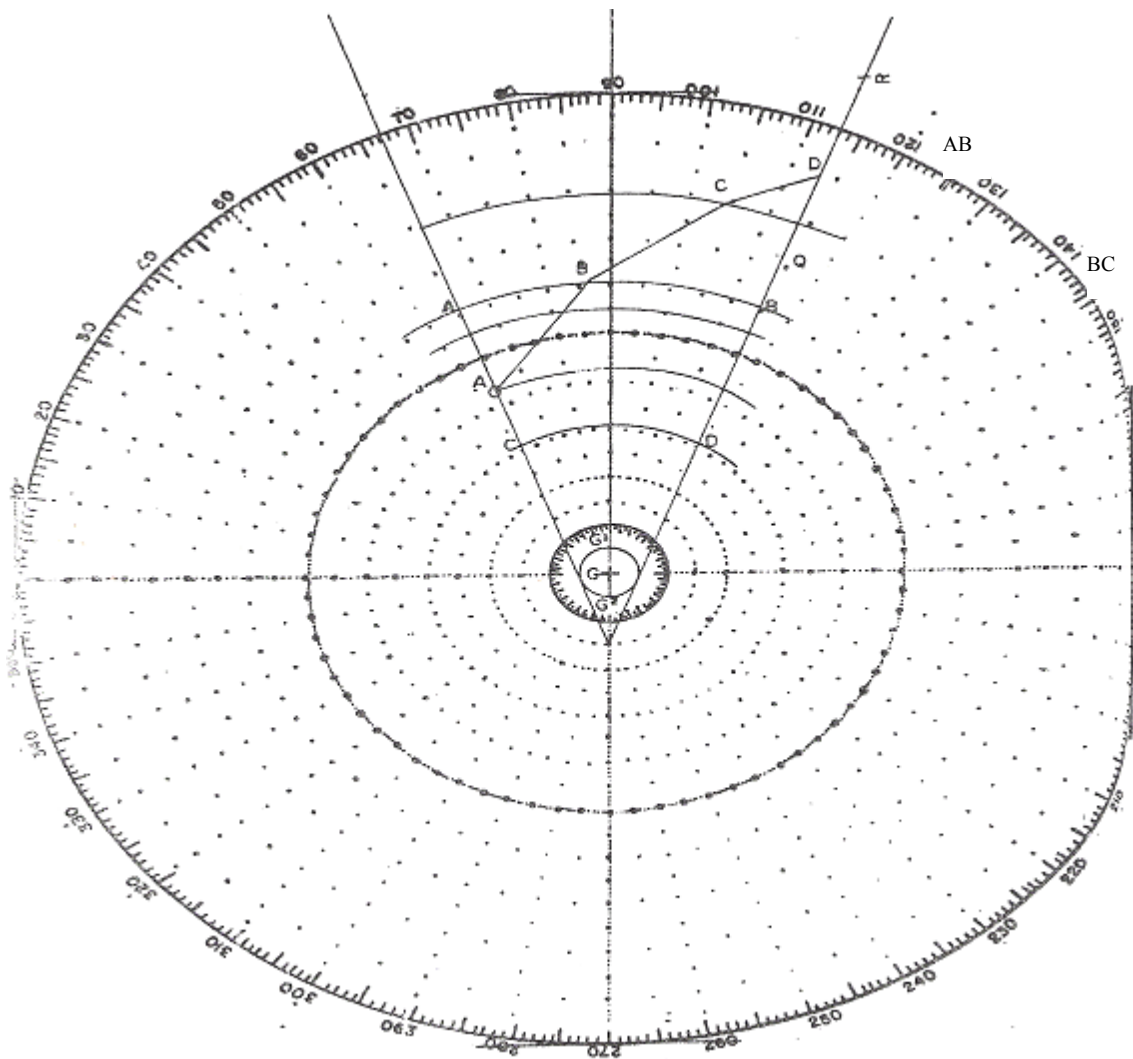
R3 .- 157°

Este rumbo es obtenido en forma gráfica por medio de la rosa de maniobras, en los cuales se navegará por una hora, recorriendo el Rumbo 3 (R3) en un tiempo de 40 min.

El tiempo a emplear durante la búsqueda es 2 hrs. 40 min. (se obtuvo con la distancia navegada sobre los rumbos de búsqueda a velocidad de exploración estos datos son enteramente gráficos y al moneto de navegar).

Nota: El frente de búsqueda se calculará de acuerdo al número de exploradores con que se cuente y la probabilidad de contacto (PC) requerida.

La forma grafica de representar los resultados antes mencionados se muestran en la Gráfica 4.0.7(a)



Gráfica 4.0.7 (a)

De igual forma que en el problema anterior se utilizara el sistema de aplicación

Con el menú de la Figura 4.0.1(a) se selecciona la búsqueda por flanco. Posteriormente se introducirán los datos requeridos para encontrar su solución.

La Figura 4.0.7(a) muestra la pantalla de búsqueda de búsqueda por flanco. La cual pide los siguientes datos:

Buque Propio

- Velocidad del buque.
- Ancho de barrido
- Probabilidad y Factor de Barrido.

Buque a localizar

- Reporte de avistamiento del buque
- Hora en que zarpa el buscador

- Distancia a la que se diviso de buque.
- Error de posición, rango de error.
- Velocidad mínima, probable y máxima que se estima.
- Tiempo de búsqueda
- Tiempo de intercepción.

busqueda_flanco

Anota los siguientes datos

Buque Propio		Buque a localizar		
Velocidad	Probabilidad	Reporte	Zarpár	MV
27 Nudos	80 [0 - 100] %	0915	1000	170 grados
Ancho de barrido	Según tablas de la normal	Distancia	Error de posición	
20 Millas	factor de barrido C= 1.28155	30 Millas	6 Millas	
con una probabilidad del: 80 %		Velocidades (Nudos)		
Si desea usar el [C] obtenido anotelo, en caso contrario deme su valor de C:		Vmin	Vprob	Vmax
1.28		15	18	21
[chechar tablas]		Tiempo de búsqueda 2. Millas		
		Tiempo de intercepción (hrs) 1.66		
BUSCAR				
Resultados Teóricos				
Derrota (S)		Tiempo muerto (Hrs.)	Nº de exploradores	
19.5781	18.46 Millas	0.75	2	
Corregida (Sc)	Pto. de intercepción (A)	Nva. Prob. Contacto		
18.46 Millas	43.38 Millas	72.138 (0 - 100) %		
Tiempo Total (Hrs)	Nuevo Total (Hrs)	Avance Max (Millas)	Avance Min (Millas)	
2.41	4.41	50.61 Sector AB	36.15 Sector CD	
Distancia al punto		Frente de exploración (F)	Nuevo Frente de Exploración (F')	
(A') 13.5	"Q" 66.15	26.46 Millas	38.46 Millas	
"R" 92.61				
[Simular] [Limpiar]				

4.0.7 (a) Búsqueda por Flanco

De igual forma se realiza el procedimiento del problema anterior. Es decir se genera el valor de C y se realiza la búsqueda generando la siguiente información:

- Derrota
- Derrota corregida, debido a que para este tipo de búsqueda el número de derrota tendrá características particulares como por ejemplo debe ser entero, lo cual modificara algunos datos.
- Rumbo
- Tiempo muerto
- Número de exploradores
- Punto de intercepción
- Nueva probabilidad de contacto

- Avance máximo y mínimo
- Distancias
- Frente de exploración y nuevo frente de exploración.

Posteriormente se genera la Simulación grafica, como se ilustra con la Figura 4.0.7(b).Donde dicha Figura describe la trayectoria que sigue este tipo de búsqueda, para una intercepción de flanco.

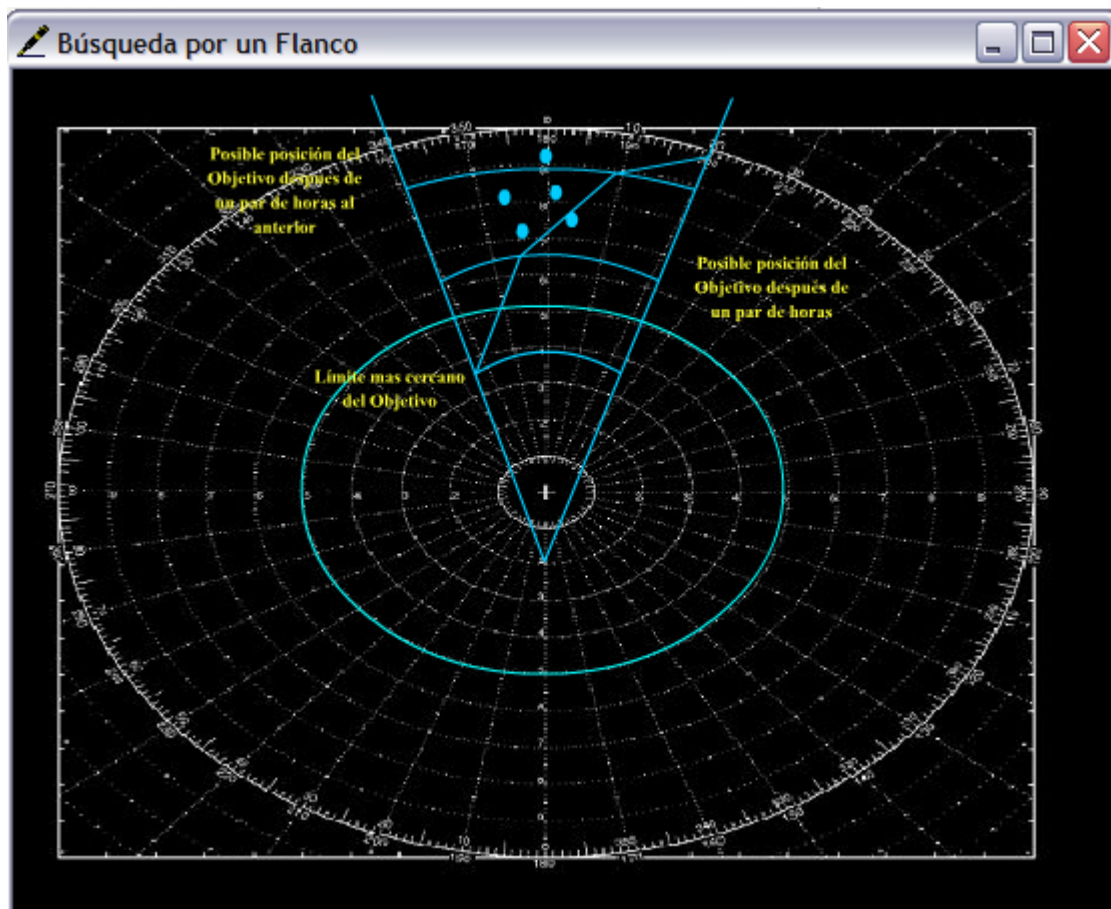


Figura 4.0.7(b) Búsqueda por un Flanco

Capítulo V

Búsqueda Asistida por Computadora

Este capítulo mostrará la utilización de métodos computacionales, para planes de búsqueda a través de mapas de probabilidad de posición de objetivos en un cierto tiempo para una elección del usuario. Estas implementaciones de metodología de búsquedas, pertenecen a una categoría importante en las tácticas de decisión ayudándonos a saber como hacer la búsqueda asistida por computadora (CAS computer-assisted search). Los programas de CAS han tenido aplicaciones de éxito en muchos ejercicios y operaciones sobre la búsqueda de guarda costas y rescate.

5.1 Generación de Números Aleatorios¹

Los métodos probabilísticos abarcan una colección de técnicas que permiten obtener soluciones de problemas matemáticos o físicos por medio de pruebas aleatorias repetidas. En la práctica, las pruebas aleatorias se sustituyen por resultados de ciertos cálculos realizados con números aleatorios.

Un sistema que maneja números aleatorios se denomina estocástico ya que funciona, sobre todo, por el azar. Las leyes de causa-efecto no explican cómo actúa el sistema de manera determinista, sino en función de probabilidades. En matemáticas de probabilidades, un proceso estocástico es un proceso aleatorio que evoluciona con el tiempo.

En estadística, un fenómeno determinístico es aquel en que se obtiene siempre el mismo resultado bajo las mismas condiciones iniciales. La relación causa-efecto se conoce en su totalidad. Por ejemplo, todos los fenómenos que siguen las leyes de la física clásica, como puede ser la caída de un cuerpo. Lo contrario de un fenómeno determinístico es un fenómeno aleatorio.

Se denomina variable aleatoria, a una variable X que puede tomar un conjunto de valores $\{x_0, x_1, x_2, \dots, x_{n-1}\}$, con probabilidades $\{p_0, p_1, p_2, \dots, p_{n-1}\}$.

Una función, variable o sistema discreto, en contraposición a continuo, si es divisible un número finito de veces. Así, el conjunto de los números naturales es un conjunto discreto. $N \rightarrow \{1, 2, 3, 4, 5, \dots\}$ Entre cada uno de los miembros del conjunto no puede haber más términos. Como se ve en los naturales se puede llegar a una sucesión indivisible de números.

5.1.1 Método Monte Carlo

Bajo el nombre de Monte Carlo o Simulación Monte Carlo se agrupan una serie de procedimientos que analizan distribuciones de variables aleatorias usando simulación de números aleatorios.

Monte Carlo da solución a una gran variedad de problemas matemáticos haciendo experimentos con muestreos estadísticos en una computadora. El método es aplicable a cualquier

¹ <http://docencia.50webs.com/simula10.htm>

tipo de problema, ya sea estocástico o determinístico (ya antes mencionados). Generalmente en estadística los modelos aleatorios se usan para simular fenómenos que poseen algún componente aleatorio. Pero en el método Monte Carlo, por otro lado, el objeto de la investigación es el objeto en sí mismo, un suceso aleatorio o pseudo-aleatorio se usa para estudiar el modelo.

La simulación de Monte Carlo también fue creada para resolver integrales que no se pueden resolver por métodos analíticos, para solucionar estas integrales se usaron números aleatorios. Posteriormente se utilizó para cualquier esquema que emplee números aleatorios, usando variables aleatorias con distribuciones de probabilidad conocidas, el cual es usado para resolver ciertos problemas estocásticos y determinísticos, donde el tiempo no juega un papel importante. La clave de la simulación Monte Carlo en este proyecto es la generación de números aleatorios.

Tras repetir n veces este experimento, dispondremos de n observaciones sobre el comportamiento del sistema, lo cual nos será de utilidad para entender el funcionamiento del mismo: obviamente, el análisis será tanto más preciso cuanto mayor sea el número n de experimentos que se lleven a cabo.

5.2 Implementación

Para poder explicar la implementación del Sistema Asistido por Computadora, el cual usa el método Monte Carlo para la generación de números aleatorios y el Teorema de Bayes para hacer el cálculo de las probabilidades a posteriori, después de la asignación de las probabilidades aleatorias a priori (método Monte Carlo); se utilizará un ejemplo simple, para su entendimiento.

Cabe mencionar que los conceptos básicos del CAS que son usados en la planeación de búsqueda para objetivos tanto en movimiento como fijos, son: discretización de mapas, construcción de multi-escenarios de una distribución a priori (mapa de probabilidad a priori) de un blanco estacionario. Actualización Bayesiana para información negativa y distribución óptima de esfuerzos de búsqueda.

Discretización de Mapas. En aplicaciones del CAS, las posiciones geográficas en una región de búsqueda son siempre mostradas por dividir la región en arreglos rectangulares de células discretas. Un simple ejemplo de un arreglo de 3×3 de tales células es mostrado en la figura 5.2(a). Ahí las células son indexadas 1, 2 y 3 en latitud y lo mismo en longitud. Estas pudieran ser, también, adecuadamente indexadas por la mitad de las latitudes o longitudes de las células.

Un programa CAS usualmente selecciona el tamaño de la célula, pero es deseable y usual dejar que el usuario cambie esta selección. Los principales factores que influyen esta selección son la certeza en ubicar el esfuerzo de búsqueda y la estimación de las probabilidades referentes a la posición. Regularmente para establecer bien las probabilidades desplegadas en un mapa, es necesario que este tenga un número realista de células. Esto puede hacerse si se promedia cada una de las células interiores con sus “vecinos” (pre-establecer), usando pesos apropiados, (mapa de probabilidad a priori) de un blanco estacionario. La distribución está dada por un número entre 0 y 1 para cada una de las células, estos números sumados deben dar 1. Cada una de los números

asignados es la probabilidad, antes de que la búsqueda empiece, es decir el blanco está en esa célula.

Construcción de multi-escenarios a priori. En la siguiente gráfica 5.2(a), dos escenarios, I y II son supuestos. Cada uno de estos escenarios es una postura estimada de lo que se espera son las causas por la que el objetivo esté en donde se estima. Asociado con cada uno de los escenarios está una distribución de la posición del objetivo que ha sido estimada desde el escenario, esta estimación se basa en la información del escenario y es la base que proporciona la posición del objetivo, ésta es denominada la derivación causal. La distribución está dada por un número entre 0 y 1 para cada una de las células, estos números sumados deben dar 1. Cada uno de los números asignados es la probabilidad, antes de que la búsqueda empiece, es decir, el blanco está en esa célula, proporcionando al escenario una cierta validez. También asociado con cada uno de los escenarios se les asigna un número entre 0 y 1, llamado el **peso del escenario**. Los pesos del escenario (en este ejemplo tenemos dos) también deben sumar 1. Cada uno de los pesos es un estimado de la probabilidad de que el escenario es válido. Usualmente se llega a estos valores consultando la opinión de expertos y pueden ser observados como “probabilidades subjetivas”.

Construcción Multiescenarios de Distribución a Priori de Posición del objetivo

(a) Escenario I
Peso = .7

Latitud Indexada	Longitud indexada		
	1	2	3
1	.1	.3	.0
2	.3	.3	.0
3	.0	.0	.0

(b) Escenario II
Peso = .3

Latitud Indexada	Longitud indexada		
	1	2	3
1	.00	.00	.00
2	.00	.25	.25
3	.00	.25	.25

Distribución a Priori de Posición del Objetivo.

Latitud Indexada	Longitud indexada		
	1	2	3
1	.07	.210	.000
2	.210	.285	.075
3	.000	.075	.075

Gráfica 5.2(a)

El escenario compuesto es obtenido por combinar las probabilidades de células de cada una de las células de cada uno de los escenarios de acuerdo a los pesos del escenario:

Para la célula (1,1) del escenario compuesto, su probabilidad es $.1 \times .7 + .00 \times .3 = .07$

Para la célula (1,2) del escenario compuesto, su probabilidad es $.3 \times .7 + .00 \times .3 = .210$

Para la célula (2,2) del escenario compuesto, su probabilidad es $.3 \times .7 + .25 \times .3 = .285$

Actualización de Información Negativa. El esfuerzo de búsqueda es aplicado al arreglo de células de 3x3, resultando una probabilidad de detección de:

Latitud Indexada	Longitud indexada		
	1	2	3
1	.0	.3	.0
2	.0	.4	.0
3	.0	.0	.0

Gráfica 5.2 (b)

Las probabilidades de detección dadas en la gráfica 5.2(b) son probabilidades condicionales, es decir, que según los planes de búsqueda, el objetivo presuntamente está, sobre las células ya mencionadas y son indicativas solamente de la calidad y cantidad del esfuerzo de búsqueda y no dicen ninguna cosa acerca de la localización del blanco.

Si se llega a suponer que el esfuerzo no fue exitoso. ¿Cuál es el nuevo mapa a posteriori?. Lo que es un hecho, es que si el objetivo no se encuentra en la células propuestas anteriormente buscadas, aumentan las probabilidades de que este en otras células. Lo anterior es una información importante y no debería ser desechada.

Pero ¿Cómo sería el ajuste del mapa de probabilidad a priori de acuerdo a la información reciente?

La respuesta es aplicar el teorema de Bayes ya mencionado en el anexo A de decisión, en la parte de probabilidad. Ahora ¿Cómo sería el ajuste del mapa de probabilidad a priori de acuerdo a la información reciente? Esto podría ser hecho en una hoja de cálculo, como sigue: (debe ser claro que el CAS lo hace de manera automática). Gráfica 5.2 (c).

[1] Célula Lat/Ing Index(i,j)	[2] Probabilidad Priori. Blanco esta en (i,j)	[3] Probabilidad de búsqueda de falla si el blanco esta en (i,j)	[4] [2]x[3]	[5]=[4]/S Probabilidad a posteriori Blanco esta en (i,j)
(1,1)	0.70	1.0	.70	.085
(1,2)	.210	.7	.147	.179
(1,3)	.000	1.0	.000	.000
(2,1)	.210	1.0	.210	.255
(2,2)	.285	.6	.171	.208
(2,3)	.075	1.0	.075	.091
(3,1)	.000	1.0	.000	.000
(3,2)	.075	1.0	.075	.091
(3,3)	.075	1.0	.075	.091
	1.000	1.0	S=.823	1.000

Gráfica 5.2 (c)

Esta Gráfica 5.2 (c) se explica de la siguiente manera: La columna [2] es la probabilidad a priori. La columna [3] es usualmente llamada la **probabilidad (likelihood)** del evento observado dado el evento inferido. La columna [4] es proporcional a la distribución a posteriori, el cual refleja observaciones del mundo real. Hasta aquí, esta información es suficiente para el usuario, por lo que, puede ser dejado en esta forma no normalizada. Pero si las probabilidades son necesarias, la columna [4] es normalizada. Esto se hace al dividir la columna [4] por su suma, resultando en la columna posterior [5]. La probabilidad a posteriori es presentada en la siguiente Gráfica 5.2(d).

Latitud Indexada	Longitud indexada		
	1	2	3
1	.085	.179	.000
2	.255	.208	.091
3	.000	.091	.091

Gráfica 5.2 (d)

Utilizando los implementos precedentes del Teorema de Bayes para esta aplicación. Para la célula (i,j) este resultado es dado por la fórmula:

$$Pp\{ B(i,j) / ND \} = P(BND(i,j) \times PA\{B(i,j)/ FN \} \dots\dots\dots(25)$$

Probabilidad a posteriori : Pp.

(B(i,j)/ ND) : blanco este en (i,j) dado que no fue detectado.

$P(BND(i,j))$: probabilidad de blanco no detectado en (i,j) .

$PA(B(i,j) / FN)$: probabilidad a priori de que el blanco este en (i,j) dado un factor de normalización.

Búsqueda Óptima Contra Objetivo Fijo. Las probabilidades en los cuadros anteriores dependen de la cantidad de esfuerzo de búsqueda que es aplicado a cada una de las células. Usualmente un planeador de búsqueda puede seleccionar entre varias distribuciones de esfuerzos, célula por célula, lo cual es preferible.

Para ilustrar esto, se supondrá que la búsqueda es por avión y se busca una lancha salvavidas, suponiéndola estacionaria. Suponiendo que la naturaleza, es tal que la probabilidad de detección acumulada a través de la búsqueda de tiempo t , $Fd(t)$, es dada por la formula de búsqueda aleatoria:

$$Fd(t) = 1 - e^{-wvt/A} \quad \text{Véase ecuación (22) del Capítulo III.}$$

Donde w es el ancho de barrido, v es la velocidad de búsqueda y A es el área de la célula buscada. Suponga $w = 30$ nm, $v = 200$ nudos y $A = 20,000$ nm². . Entonces: Véase ecuación (22) del Capítulo III.

$$\begin{aligned} Fd(t) &= 1 - e^{-wvt/A} = 1 - e^{-(30)(200)t/(20,000)} \\ Fd(t) &= 1 - e^{-0.3t} \end{aligned}$$

(puede ser que w , y de acuerdo al coeficiente 0.3 en $Fd(t)$, cambia de una célula a la otra, pero suponga en este caso que esto no sucede).

Véase la Gráfica 5.2(a), la célula (2,2) debería ser objeto del esfuerzo inicial por ser ésta la de mayor probabilidad, 0.285. Pero ¿Cuánto tiempo debería la búsqueda permanecer en (2,2) antes de poner el esfuerzo en (1,2) y (2,1), las cuáles tienen las segundas probabilidades a priori más grandes de contener el blanco 21? Por lo que no se puede aplicar el algoritmo Bayesiano para encontrar el valor de t por eso se calcula la probabilidad a posteriori en (2,2) para .21. Sin embargo cuando la probabilidad a posteriori cae en (2,2), esta se eleva en (1,2) y (2,1). Véase Gráfica 5.2(d). La solución, desde luego, es encontrar la t donde estas caídas y subidas de probabilidades se encuentran, Nótese que la probabilidad de falla acumulada es $\exp(-0.3t)$ y poniendo las probabilidades a posteriori de las células (2,2) y (1,2) e igualando a cada una, se tiene que:

$$0.285e^{-0.3t/S} = 0.21/S$$

Y resolviendo para t , entonces:

$$t = \ln(0.21/0.285)/-0.3 = 1.02 \text{ hrs.}$$

Esto es que después de 1.02 horas el esfuerzo debería ser dividido entre (2,2), (1,2) y (2,1), puesto que las tres células tienen la misma probabilidad (la cual no ha sido recalculada hasta ese momento) en ese punto.

Este procedimiento puede continuar hasta llegar a que todas las células que inicialmente tenían probabilidad diferente de cero, de contener al blanco, ahora tengan la misma probabilidad de encontrar el blanco, y en consecuencia, la búsqueda subsiguiente es dividida igualmente entre ellas.

Búsqueda Óptima Contra Objetivo Fijo. El algoritmo anterior puede ser cambiado como un algoritmo en la forma más general, como sigue:

Sea C un conjunto finito de células, una de las cuáles contiene el blanco. Una cantidad de esfuerzo de búsqueda es disponible, el cual puede ser dividido entre las células en C , por el “buscador”.

Suponga una efectividad exponencial de esfuerzo de búsqueda, en el sentido de que hay un $\beta > 0$ tal que si z es una cantidad de esfuerzo de búsqueda aplicada a la célula que contienen el blanco, entonces $1 - \exp(-\beta z)$ es la probabilidad de detección que resulta de z . (las dimensiones de z y β deben ser de tal manera que βz sean unidimensionales). Este esfuerzo disponible inicial es totalmente aplicado en una sucesión de pasos de aplicación.

El paso inicial de aplicación empieza con el esfuerzo total resultante z y para c en C , la probabilidad $p(c)$ en ese punto donde contiene el blanco, puede justamente ser supuesto que $p(C) > 0$, de ahí que si $p(C) = 0$, ningún esfuerzo de búsqueda debería ser aplicado a c . Si cuando el proceso empieza, todas las células tienen el mismo valor $p(c)$, entonces dividamos z igual entre las otras. De otra manera, seleccionemos C_1 y C_2 en C tal que $p(c_1)$ y $p(c_2)$ son respectivamente el valor más alto y el segundo más alto de p . Entonces para cada c en C cuyo valor p es $p(c_1)$ aplicando a la cantidad de esfuerzo de búsqueda:

$$t = -(1/\beta) \ln(p(c_2)/p(c_1)) \dots\dots(26)$$

Al menos esto agotaría a z , y a ninguna otra célula. Si el agotamiento va a ocurrir, entonces en lugar de ello, se divide z por igual entre las células con probabilidad $p(c_1)$ y la distribución es completa. Ocurra o no el agotamiento, se calcula:

$$S = \sum p(c) e^{-\beta x(c)} \dots\dots(27)$$

Donde cada una de las c , $x(c)$ es el esfuerzo aplicado a c en este paso de aplicación. Entonces S es la probabilidad de no-detección para este paso de aplicación. Si el blanco no ha sido encontrado y el esfuerzo aún queda como, $p(c)\exp(-\beta x(c))$ la cual, es la probabilidad que controla o da las restricciones en el principio del próximo paso (S es el factor de normalización). Se repite el procedimiento hasta que el esfuerzo se agote. Este resultado es una distribución óptima. La probabilidad de falla para el procedimiento entero es el producto de los factores S sobre el total de los pasos de aplicación. Si la probabilidad de falla del procedimiento total no es necesaria, no hay necesidad de que los números de normalización proporcionales a las probabilidades condicionantes sean suficientes para producir la distribución.

Nótese que este algoritmo es “miope”, esto es, uno siempre busca en la célula(s) de mayor probabilidad corriente. Esto optimiza el esfuerzo disponible sin importar que un esfuerzo adicional pueda llegar a estar disponible después. Por ejemplo, suponga que un planeador tuviera inicialmente distribuidas 4 unidades de esfuerzo de búsqueda y planeado en común acuerdo con el método anterior. Entonces por disposición de equipo, al planeador se le permite 3 unidades adicionales de esfuerzo. Puede, entonces ser deseable que las primeras 4 unidades que han sido utilizadas pudieran ser utilizadas de manera diferente, a la luz de tener un total de 7 unidades disponibles. La respuesta es NO, la búsqueda miope es óptima. Esta aseveración depende mucho de que el blanco es estacionario. Cuando el blanco esta en movimiento, la búsqueda miope no es óptima.

5.2.1 Simulación del CAS (Búsqueda Asistida por Computadora).

Por sus siglas en ingles (CAS computer-assisted search). Es importante mencionar que el presente trabajo, solamente se diseño para simular la búsqueda de objetos estacionarios.

La probabilidad a priori fue generada con la aplicación del método Monte Carlo; el cual proporciona la generación de las probabilidades de cada una de las celdas de posible búsqueda.

La siguiente Figura 5.2.1(a) muestra la pantalla del simulador de búsqueda.

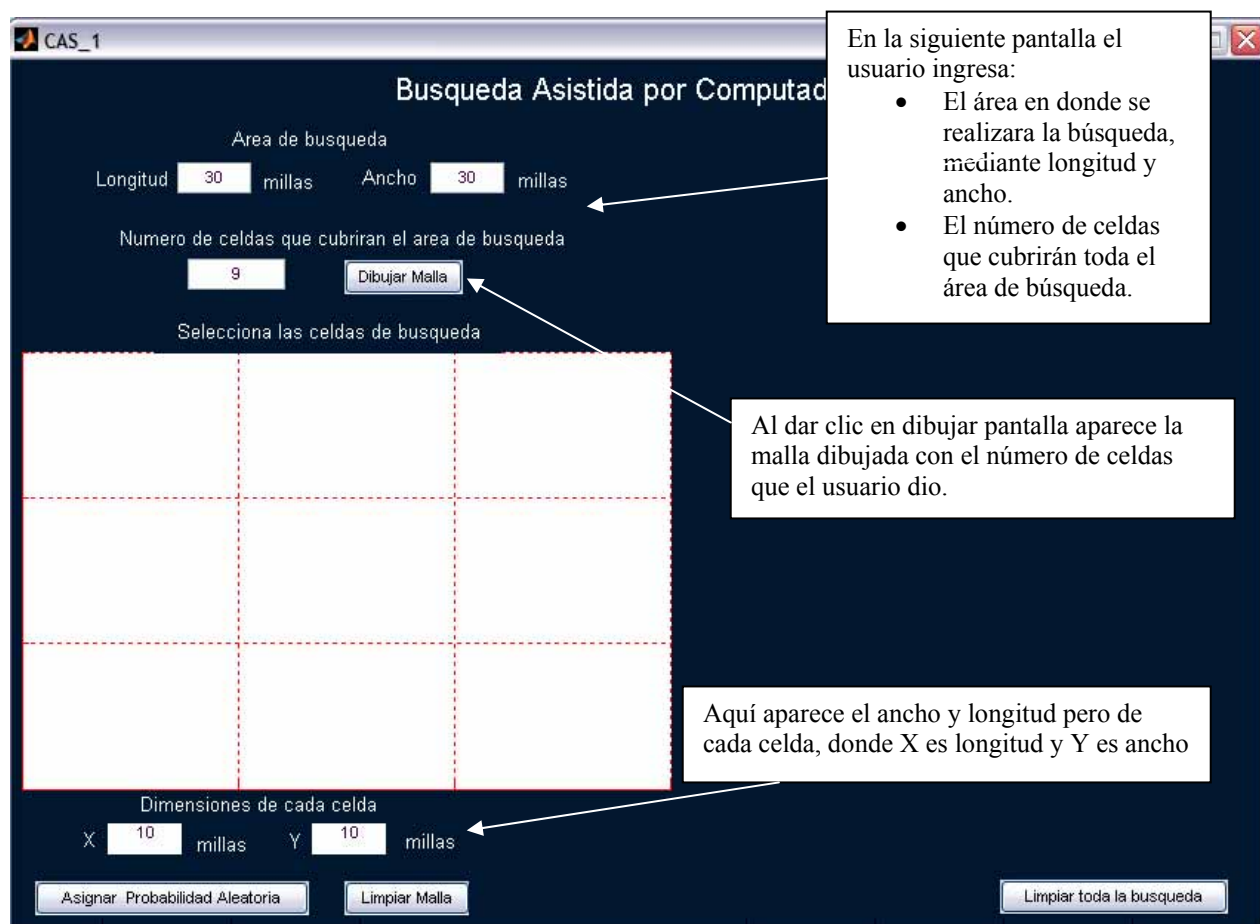


Figura 5.2.1(a)

El usuario selecciona el número de celdas que contendrá la malla de búsqueda (esta malla representa el área total donde se realizara la búsqueda).

El usuario podrá seleccionar las zonas de búsqueda, como muestra la Figura 5.2.1(b).

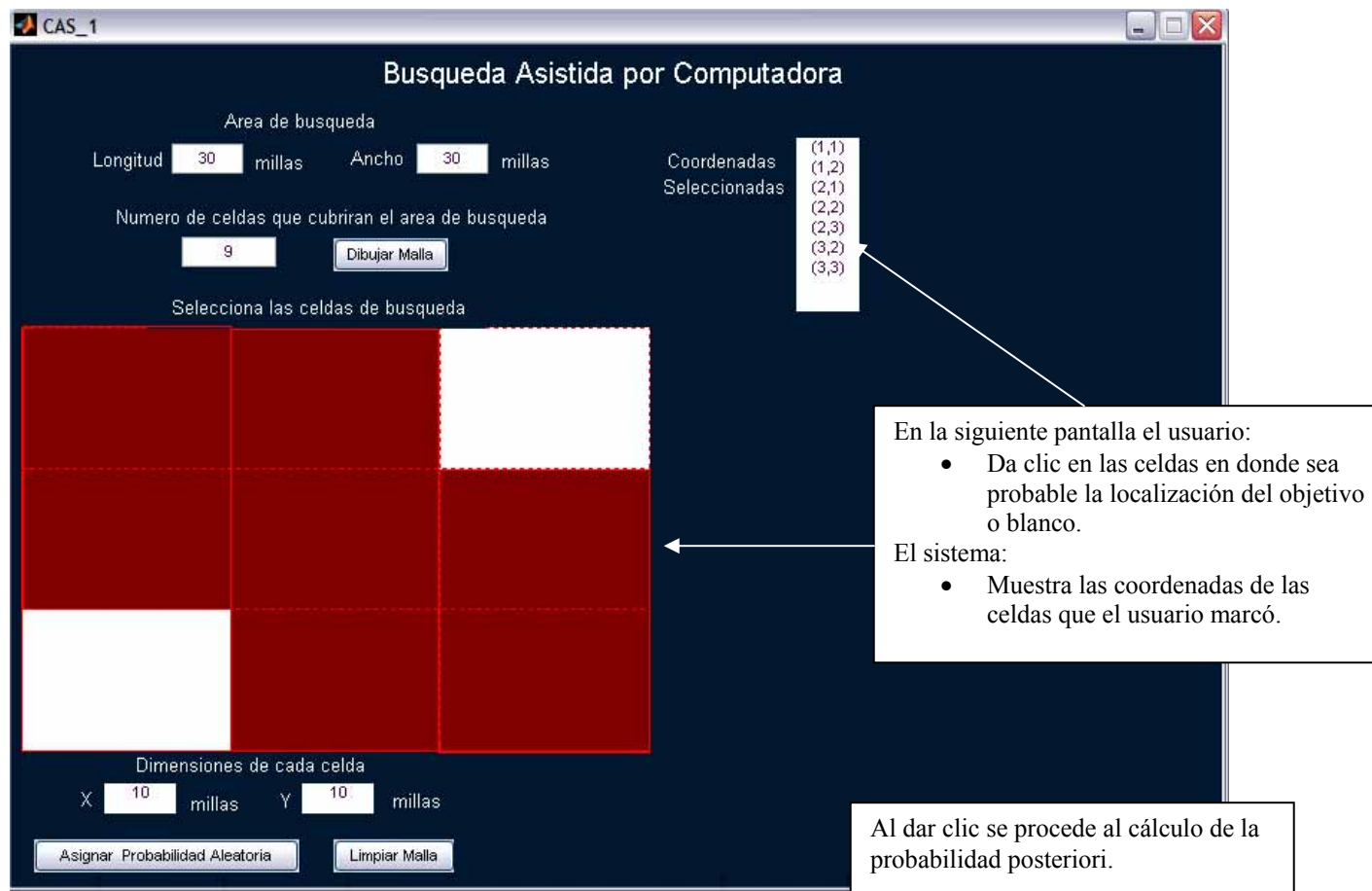


Figura 5.2.1(b)

Dando click en el boton de asignación de probabilidad aleatoria se generara la probabilidad de las celdas seleccionadas (Figura 5.2.1(c)). Esta es la probabilidad a priori, la cual es generada por el método de Monte Carlo o directamente por el usuario.

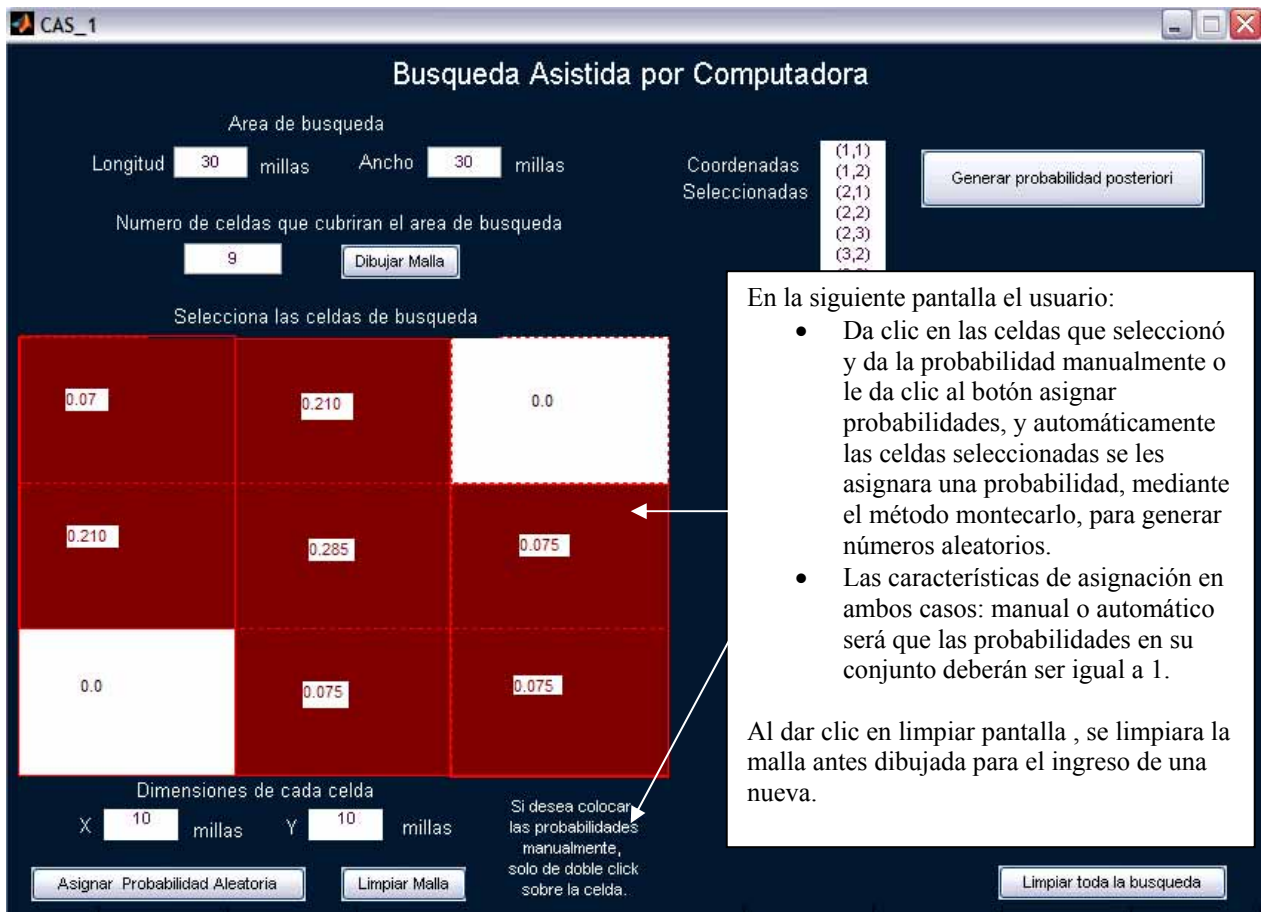


Figura 5.2.1(c)

Después se procede al cálculo de la probabilidad posteriori, en donde se toman en cuenta las celdas de probabilidad mayor y la subsecuente a la mayor, de tal modo que al aplicar el algoritmo, el sistema sabrá reconocer probabilidades para comenzar la búsqueda. Esto se hará en automático, con la característica de que ya con la malla de probabilidad posteriori, se pedirán los datos de ancho de barrido, velocidad, y más parámetros para empezar con la búsqueda.

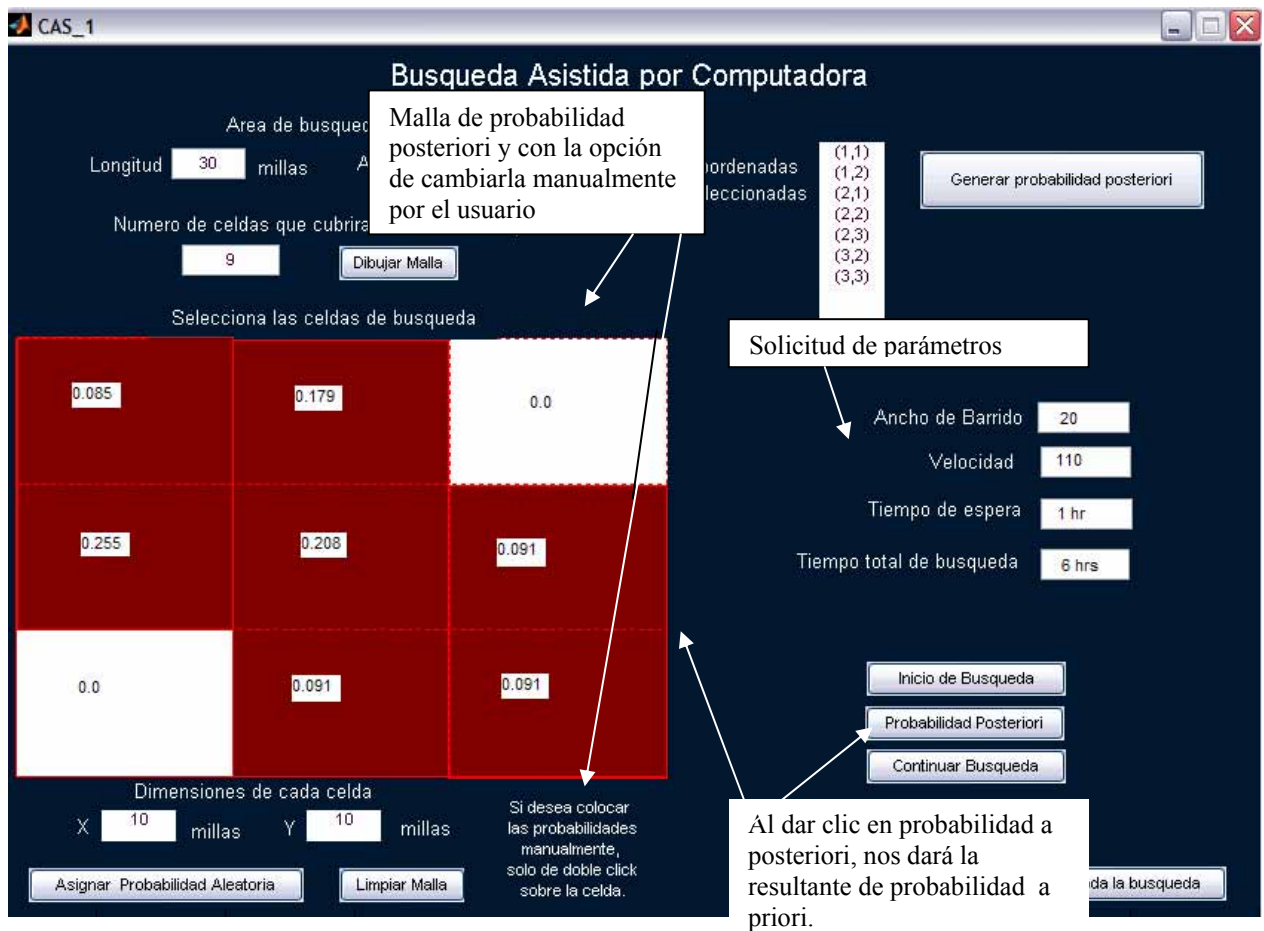


Figura 5.2.1.(d)

ya teniendo la probabilidad posteriori se inicia la búsqueda, conociendo el tiempo total de búsqueda, el cual nos da la pauta para determinar el fin de la búsqueda .

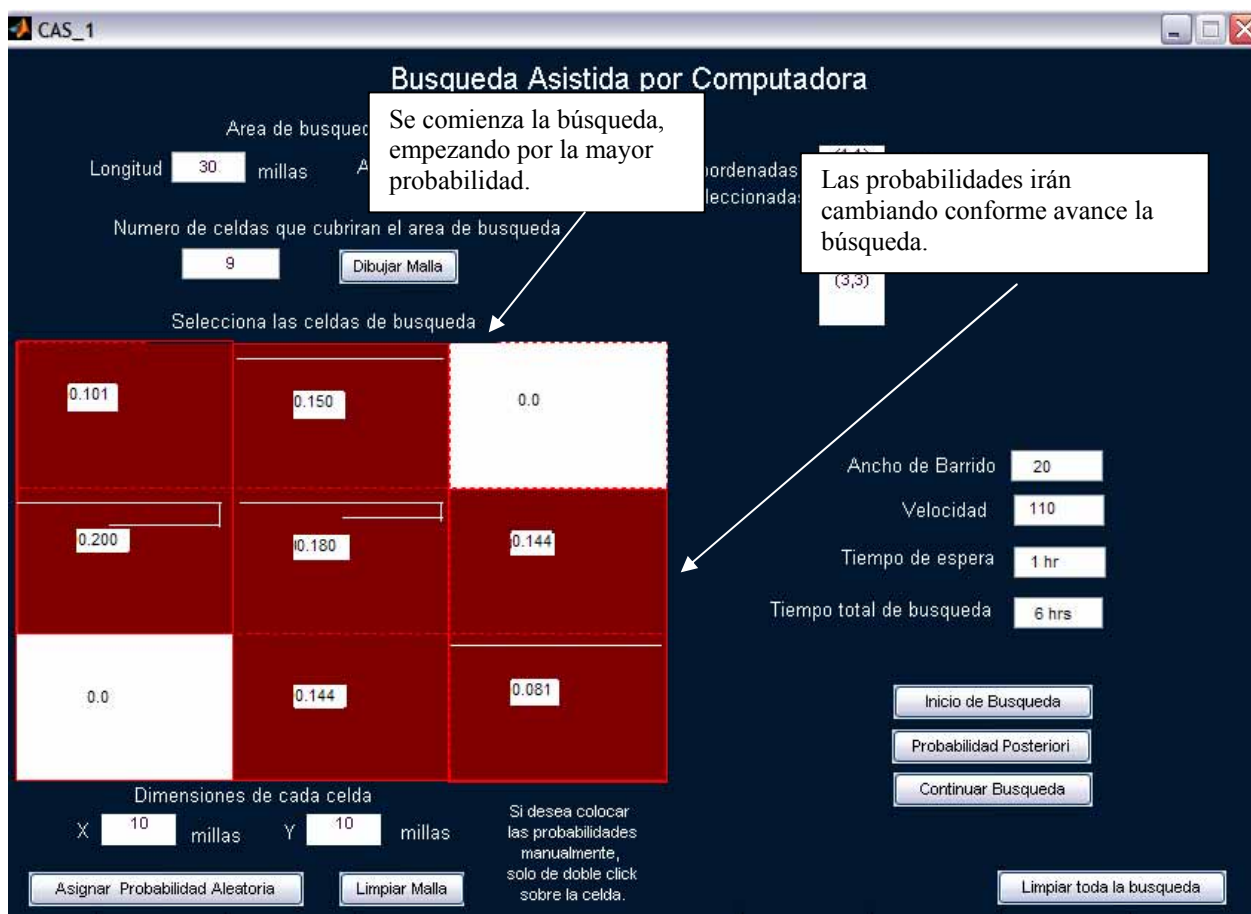


Figura 5.2.1(e)

Las probabilidades a priori antes vistas en la Figura 5.2.1(c) se guardan en un registro, para proseguir con el calculo de la posteriori, Figuras 5.2.1(d) y 5.2.1(e), hasta llegar al final de la búsqueda según el tiempo total.

Conclusiones

El análisis de operaciones, es por muchos aspectos, de gran utilidad para los directivos responsables de tomar una decisión, ya que provee diferentes alternativas para la posible solución de algún problema, basándose en métodos de investigación y conocimientos matemáticos, para trabajar en mancuerna con el directivo, ya que una decisión a la ligera podría traer consecuencias graves, si es que no se da la importancia que esta lleva.

Para la toma de decisiones la experiencia del directivo es de gran importancia, por que puede marcar la pauta para sopesar las alternativas ofrecidas e inclinarse hacia una en específico. Es aquí, donde se conjuga el análisis de operaciones, el directivo y el apoyo en una herramienta computacional. El papel que juega el cómputo en esta situación, es la creación de una herramienta eficiente, de fácil manejo, y sobre todo una rápida respuesta, cuando el tiempo es una limitante para la toma de decisiones, y no es de sobra decir, que dicha herramienta se basa en métodos del análisis de operaciones, para que en su conjunto, ofrezcan al directivo, la diferentes alternativas que lo lleven a tomar una decisión.

El sistema de búsqueda asistida por computadora, para este proyecto en específico fue de “búsqueda y detección en superficie”, deja como aprendizaje que no sólo en al ámbito naval, el análisis de operaciones puede ser utilizado, sino que, su aplicación llega a cualquier situación o campo científico en donde el análisis sea necesitado, junto con su mancuerna computacional para un mejor desempeño, pues en algunas ocasiones, los métodos empleados requieren hacer cálculos matemáticos engorrosos y numerosos, que si son hechos a mano, quitan tiempo y desgastan al analista, por ello la ayuda de una herramienta de cómputo es esencial.

En la actualidad, los métodos descritos en este trabajo son empleados para búsqueda y detección en cualquier marina del mundo, además de que la búsqueda no se limita a buques o posibles amenazas, sino que puede ser empleada para balsas que han naufragado, e incluso personas a la deriva, por lo que el desarrollo de una herramienta de cómputo, abre puertas para proyectos más grandes.

La proyección que puede lograr este sistema en específico, es que la búsqueda no se limita en la mar, pues la búsqueda para hacer alguna detección podrá hacerse sobre cualquier área lisa o concreto, además de que una búsqueda no queda solo en la superficie, sino que puede ser aérea, o submarina, acoplando a cada tipo de búsqueda los artefactos de medición que estos necesitan. Y no sería presuntuoso decir, que este sistema, sería el comienzo de algo que pudiera llegar a ser un simulador práctico y funcional para búsqueda y detección en el ámbito naval.

ANEXO A

Probabilidad

La teoría de probabilidad es una rama de las matemáticas con más uso en la investigación de operaciones. Casi todos los resultados de operaciones de guerra involucran elementos de probabilidad, usualmente para tener un mayor alcance, de modo que sólo cuando un número de resultados de operaciones similares son revisados nos da una evidencia regular. Esto resulta importante para saber el grado por cada operación individual, el cual difiere un poco de la media esperada, como se sabe la media depende de las variables involucradas. En la investigación de operaciones los datos muchas veces son vagos y fragmentados, esto es necesario para habilitar la estimación que probablemente será la siguiente operación que mostrará características similares que se analizaron en esta. La probabilidad entra en muchos problemas analíticos y en todos los problemas estadísticos.

Conceptos Fundamentales

La teoría de la probabilidad estudia el problema de un experimento que se realiza varias veces; con diversos resultados posibles; a este conjunto de resultados posibles se le llama *espacio muestral o espacio de eventos* y a cada experimento se le llama *evento*. Y a cada resultado o evento del experimento, se le asocia un número llamado *probabilidad*.

Varios teoremas de probabilidad son obvios desde su definición: *La probabilidad de que algún resultado de una serie de resultados sea obtenido, es la suma de las probabilidades de cada uno de esos resultados; y la suma de probabilidades de todos los resultados es la unidad; si p es la probabilidad de un resultado, la probabilidad de que éste no sea el resultado es $1 - p$.*

Eventos Mutuamente Excluyentes. Dos eventos A y B son mutuamente excluyentes o disjuntos si $A \cap B = \emptyset$, esto es, si A y B no tienen elementos en común.

Probabilidad Axiomática. Dado un espacio muestral S y la familia R de eventos en S, una función de probabilidad asociada a cada evento A en R un número real $P(A)$, la probabilidad del evento A, tal que los siguientes axiomas sean verdaderos:

- 1.- $P(A) \geq 0$, para todo A.
- 2.- $P(S) = 1$
- 3.- Si existe algún conjunto contable de eventos $\{A_1, A_2, \dots, A_k\}$
y si estos eventos son todos mutuamente exclusivos, entonces:
$$P(A_1 \cup A_2, \dots, \cup A_k)$$

En resumen la probabilidad de un evento A, es la suma de los pesos de todos los pjn tos muestrales de A por tanto, $0 \leq P(A) \leq 1$, $P(\emptyset) = 0$ y $P(S) = 1$

Además de que la probabilidad de la unión de eventos mutuamente exclusivos es igual a la suma de sus probabilidades.

Definición Clásica de Probabilidad. Si un evento A puede descomponerse como la unión de $N(A)$ eventos pertenecientes a una partición constituida por N eventos mutuamente exclusivos e igualmente posibles, entonces la probabilidad $P(A)$ del evento A, esta dado por:

$$P(A) = \frac{N(A)}{N} = \frac{\text{Número de eventos en A}}{\text{Número total de eventos en S}}$$

Probabilidad Condicional. Sea un experimento E con espacio S y un evento B con probabilidad asociada diferente de 0. Es posible definir una nueva probabilidad para todos los eventos A del experimento, que se llamará probabilidad condicional dado B.

$$P(A / B) = \frac{P(A \cap B)}{P(B)}$$

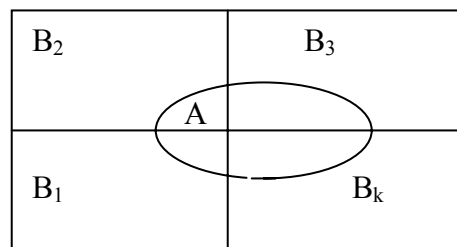
Probabilidad Total. Si se tienen n eventos mutuamente exclusivos, $A_1, A_2, \dots, A_n$ cuya suma es igual a S y un evento B arbitrario el teorema permite afirmar que:

$$P(B) = P(B/A) ((P(A))) = \sum_{i=1}^n P(B/A_i) P(A_i)$$

Densidad de Probabilidad Normal. Esta densidad es la de las sumas (o promedios) de números grandes de variables aleatorias.

Con cierta falta de rigor se considera que la densidad normal constituye un modelo adecuado para muchas situaciones en las que existe multiplicidad de efectos. (por ejemplo. La trayectoria de una bala en un disparo). Lo anterior puede ser cierto por que requiere generalmente justificación a posteriori por experimentación.

Teorema de Bayes



El teorema de bayes sirve para conocer la probabilidad de una partición, si los eventos $B_1, B_2, B_3, \dots, B_k$ constituyen una división del espacio muestral, y donde sus probabilidades son distintas de cero, entonces para cualquier evento A su probabilidad esta dada por:

$$P(A_i / B) = \frac{P(B / A_i) P(A_i)}{\sum_{i=1}^n P(B / A_i) P(A_i)} \quad \text{probabilidad total}$$

Se usa generalmente la terminología siguiente:

$P(B / A_i)$ = “ Probabilidades condicionales”

$P(A_i)$ = “Probabilidades a priori “

$P(A_i / B)$ = “Probabilidades a posteriori”

ANEXO B

El siguiente anexo contiene el código fuente de toda la aplicación.

Para el desarrollo de las pantallas en Matlab se hizo mediante una herramienta llamada guide en matlab, compuesta de herramientas para el desarrollo de interfaces graficas y la creación automática de código para esos objetos grafico, como cuadros de texto, botones, etc. De la mano le sigue un archivo con extensión .m, el cual, es el código de programación que se presenta:

Pantalla de entrada: busqueda_asistida, muestra el menú de los tipos de búsqueda:

```
function varargout = busqueda_asistida(varargin)
% BUSQUEDA_ASISTIDA M-file for busqueda_asistida.fig
%     BUSQUEDA_ASISTIDA, by itself, creates a new
%     BUSQUEDA_ASISTIDA or raises the existing
%     singleton*.
%
%     H = BUSQUEDA_ASISTIDA returns the handle to a new
%     BUSQUEDA_ASISTIDA or the handle to
%     the existing singleton*.
%
%     BUSQUEDA_ASISTIDA('CALLBACK',hObject,eventData,handl
%     es,...) calls the local
%     function named CALLBACK in
%     BUSQUEDA_ASISTIDA.M with the given input arguments.
%
%     BUSQUEDA_ASISTIDA('Property','Value',...) creates a new
%     BUSQUEDA_ASISTIDA or raises the
%     existing singleton*. Starting from the left, property value
%     pairs are
%     applied to the GUI before
%     busqueda_asistida_OpeningFunction gets called. An
%     unrecognized property name or invalid value makes property
%     application
%     stop. All inputs are passed to busqueda_asistida_OpeningFcn
%     via varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI
%     allows only one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help
busqueda_asistida

% Last Modified by GUIDE v2.5 08-May-2007 21:40:12

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @busqueda_asistida_OpeningFcn,
    ...
    'gui_OutputFcn', @busqueda_asistida_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before busqueda_asistida is made visible.
function busqueda_asistida_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to busqueda_asistida (see
VARARGIN)

% Choose default command line output for busqueda_asistida
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes busqueda_asistida wait for user response (see
UIRESUME)
% uiwait(handles.figure1);
%handles.current_data=handles.rectangular;

% --- Outputs from this function are returned to the command line.
function varargout = busqueda_asistida_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%Colocar Imagen de fondo
fondo = imread('C:\MATLAB7\work\pedras.jpg'); %Leer imagen
axes(handles.fondo); %Carga la imagen en fondo
axis off;
imshow(fondo); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

%Títulos sobre imagen
%Titulo
text(40,60,'Busqueda Asistida por
Computadora','Fontname','Arial','FontSize',20,'Fontangle','Italic','Fo
ntweight','Bold','color',[1 1 1]);
%subtitulo1
text(110,130,'Tipos de Búsqueda','Fontname','Comic
SansMS','Fontangle','Italic','Fontweight','Bold','FontSize',17,'color',[
1 1 1]);
%subtitulo2
```

```

text(60,180,'Aleatoria','Fontname','Comic
SansMS','Fontangle','Italic','Fontweight','Bold','FontSize',17,'color',[
1 1 1]);
%subtitulo3
text(210,180,'Sistematica','Fontname','Comic
SansMS','Fontangle','Italic','Fontweight','Bold','FontSize',17,'color',[
1 1 1]);
%acaban titulos del programa

% -----

% --- Executes on selection change in aleatoria.
function aleatoria_Callback(hObject, eventdata, handles)
% hObject handle to aleatoria (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns aleatoria contents
as cell array
% contents{get(hObject,'Value')} returns selected item from
aleatoria
%-----
val=get(hObject,'Value');
str=get(hObject,'String');
switch str {val}
case 'rectangular'
clc,rectangular; %clc me cierra la ventana anterior y abre mi nuevo
programa de rectangular
%openfig('rectangular.fig');
case 'expansion cuadrada'%clc me cierra la ventana anterior y abre
mi nuevo programa de expansion
clc,expansion;
end
guidata(hObject,handles);

%-----
% --- Executes during object creation, after setting all properties.
function aleatoria_CreateFcn(hObject, eventdata, handles)
% hObject handle to aleatoria (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

% Hint: popupmenu controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on selection change in sistematica.
function sistematica_Callback(hObject, eventdata, handles)
% hObject handle to sistematica (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns sistematica
contents as cell array
% contents{get(hObject,'Value')} returns selected item from
sistematica
val=get(hObject,'Value');
str=get(hObject,'String');
switch str {val}

```

```

case 'sector_base_fija'
clc, sector_base_fija; %clc me cierra la ventana anterior y abre mi
nuevo programa de sector con base fija
case 'sector_base_movil'
clc, sector_base_movil; %clc me cierra la ventana anterior y abre
mi nuevo programa
case 'busqueda_por_flanco'
clc, busqueda_flanco; %clc me cierra la ventana anterior y abre mi
nuevo programa
case 'busqueda_por_delante'
clc, busqueda_delante; %clc me cierra la ventana anterior y abre mi
nuevo programa
case 'busqueda_por_detras'
clc, intercepcion_detras; %clc me cierra la ventana anterior y abre
mi nuevo programa
end
guidata(hObject,handles);

% --- Executes during object creation, after setting all properties.
function sistematica_CreateFcn(hObject, eventdata, handles)
% hObject handle to sistematica (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

% Hint: popupmenu controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on selection change in sistematica.
function sisitematica_Callback(hObject, eventdata, handles)
% hObject handle to sistematica (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns sistematica
contents as cell array
% contents{get(hObject,'Value')} returns selected item from
sistematica

% --- Executes during object creation, after setting all properties.
function sisitematica_CreateFcn(hObject, eventdata, handles)
% hObject handle to sistematica (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

% Hint: popupmenu controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

Pantalla de búsqueda rectangular:

```
function varargout = rectangular(varargin)
% RECTANGULAR M-file for rectangular.fig
%   RECTANGULAR, by itself, creates a new RECTANGULAR
%   or raises the existing
%   singleton*.
%
%   H = RECTANGULAR returns the handle to a new
%   RECTANGULAR or the handle to
%   the existing singleton*.
%
%   RECTANGULAR('CALLBACK',hObject,eventData,handles,...)
%   calls the local
%   function named CALLBACK in RECTANGULAR.M with
%   the given input arguments.
%
%   RECTANGULAR('Property','Value',...) creates a new
%   RECTANGULAR or raises the
%   existing singleton*. Starting from the left, property value
%   pairs are
%   applied to the GUI before rectangular_OpeningFunction gets
%   called. An
%   unrecognized property name or invalid value makes property
%   application
%   stop. All inputs are passed to rectangular_OpeningFcn via
%   varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI
%   allows only one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help rectangular

% Last Modified by GUIDE v2.5 02-Jun-2007 14:39:35

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @rectangular_OpeningFcn, ...
    'gui_OutputFcn',  @rectangular_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before rectangular is made visible.
function rectangular_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% varargin command line arguments to rectangular (see
VARARGIN)

% Choose default command line output for rectangular
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

initialize_gui(hObject, handles, false);
% --- Outputs from this function are returned to the command line.
function varargout = rectangular_OutputFcn(hObject, eventdata,
handles)
% varargin cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%Colocar Imagen de piedra
piedra = imread('C:\MATLAB7\work\pedras.jpg'); %Leer imagen
axes(handles.piedra); %Carga la imagen en fondo
axis off;
imshow(piedra); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

%titulos para los cuadros de texto
text(115,15,'Anota los siguientes
datos','Fontname','Arial','FontSize',12,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(25,35,'Ancho de barrido
(millas)','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(129,35,'Area de busqueda
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(195,50,'Latitud
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(195,75,'Altitud
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(260,35,'Velocidad
(Nudos)','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(30,67,'Probabilidad [0-100]
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','No
rml','color',[1 1 1]);
text(30,97,'Según tablas de la normal
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(30,118,'con una probabilidad del:
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(69,127,'%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fo
ntweight','Normal','color',[1 1 1]);
text(175,97,'Si desea usar el
[C]','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(175,103,'obtenido,anotelo:','Fontname','Arial','FontSize',8,'Font
angle','Italic','Fontweight','Normal','color',[1 1 1]);
text(175,109,'en caso contrario
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
```

```

text(175,115,'deme su valor de C:
','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(255,67,'Numero de
Exploradores','Fontname','Arial','FontSize',8,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(115,180,'Resultados
Teóricos','Fontname','Arial','FontSize',12,'Fontangle','Italic','Fontwe
ight','Bold','color',[1 1 1]);
text(53,210,'Derrota ( S )
Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight
','Normal','color',[1 1 1]);
text(195,203,'Número de
Barridos','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(160,218,'Giros
Simultaneos','Fontname','Arial','FontSize',8,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(290,218,'Barridos','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(160,235,'Giros de
Búsqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(290,235,'Barridos','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(35,260,'Distancia Recorrida por los
Exploradores','Fontname','Arial','FontSize',8,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(235,260,'Tiempo que Dura la
Búsqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(23,275,'Giros','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(15,282,'Simultaneos','Fontname','Arial','FontSize',8,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(23,305,'Giros','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(15,311,'De
Búsqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(130,280,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(131,310,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(210,275,'Giros','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(205,282,'Simultaneos','Fontname','Arial','FontSize',8,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(210,305,'Giros','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(205,311,'De
Búsqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(310,280,'Hrs','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(310,310,'Hrs','Fontname','Arial','FontSize',8,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);

```

```

function anchobarrido_Callback(hObject, eventdata, handles)
% hObject handle to anchobarrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of anchobarrido as
text
% str2double(get(hObject,'String')) returns contents of
anchobarrido as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado

```

```

anchobarrido = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(anchobarrido)
set(hObject,'String', 0);
errorldg('Solo debes introducir numeros','Error');
end
handles.rect.anchobarrido=anchobarrido; %Almacenar en puntero
handles pero del anchobarrido
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function anchobarrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to anchobarrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function latitud_Callback(hObject, eventdata, handles)
% hObject handle to latitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of latitud as text
% str2double(get(hObject,'String')) returns contents of latitud
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
latitud = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(latitud)
set(hObject,'String', 0);
errorldg('Solo debes introducir numeros','Error');
end
handles.rect.latitud=latitud; %Almacenar en puntero handles pero
de latitud
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function latitud_CreateFcn(hObject, eventdata, handles)
% hObject handle to latitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));

```

end

```
function altitud_Callback(hObject, eventdata, handles)
% hObject handle to altitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of altitud as text
% str2double(get(hObject,'String')) returns contents of altitud
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
altitud = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(altitud)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.rect.altitud=altitud; %Almacenar en puntero handles pero
de altitud
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function altitud_CreateFcn(hObject, eventdata, handles)
% hObject handle to altitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad=str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.rect.probabilidad=probabilidad; %Almacenar en puntero
handles pero de busqueda
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
```

```
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
%----- VARIABLES PARA DETERMINAR DERROTAS (S) -----
-----
```

```
Z=handles.rect.z;
W=handles.rect.anchobarrido;
Sd=int8(1.253*(W/Z));
set(handles.derrota,'String',Sd);
S = char(Sd);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN -----
```

```
%-----VARIABLES PARA DETERMINAR NUMERO DE
BARRIDOS EN GIROS SIMULTANEOS Y DE BÚSQUEDA-----
-----
```

```
if handles.rect.altitud<handles.rect.latitud
```

```
    LM=handles.rect.altitud;
    %set(handles.simultaneo,'String',LM);
else
    LM=handles.rect.latitud;
    %set(handles.simultaneo,'String',LM);
end
set(handles.simultaneo,'String',LM);
N = handles.rect.exploradores;
BARRIDOS= LM / ( N * S );
set(handles.simultaneo,'String',BARRIDOS);
set(handles.busqueda,'String',BARRIDOS);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----
```

```
%-----VARIABLES PARA DETERMINAR DIST.
RRECORRIDA EN GIROS SIMULTANEOS-----
if handles.rect.altitud>handles.rect.latitud
```

```
    LMA=handles.rect.altitud;
    %set(handles.simultaneo,'String',LM);
else
    LMA=handles.rect.latitud;
    %set(handles.simultaneo,'String',LM);
end
```

```
DISTANCIA= [(BARRIDOS*(LMA - S )) + ((BARRIDOS - 1)*
(N * S ))+( S / 2)];
set(handles.distsimultaneos,'string',DISTANCIA);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----
```

```
%-----VARIABLES PARA DETERMINAR DIST.
RECORRIDA EN GIROS DE BUSQUEDA---
if handles.rect.altitud>handles.rect.latitud
```

```
    LMA=handles.rect.altitud;
    %set(handles.simultaneo,'String',LM);
else
    LMA=handles.rect.latitud;
    %set(handles.simultaneo,'String',LM);
end
```

```
DISTANCIAB= [[BARRIDOS*LMA] - S/2];
set(handles.disbusqueda,'String',DISTANCIAB);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----
```

```
%-----VARIABLES PARA DETERMINAR TIEMPO DE
BUSQUEDA SIMULTANEO-----
V=handles.rect.velocidad;
TIEMPO1= DISTANCIA/V ;
set(handles.hrsimultaneo,'String',TIEMPO1);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----
```

```
%-----VARIABLES PARA DETERMINAR TIEMPO DE
BUSQUEDA SIMULTANEO-----
V=handles.rect.velocidad;
TIEMPO2= DISTANCIA/V ;
set(handles.hrsbusqueda,'String',TIEMPO2);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----
```

```
function velocidad_Callback(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velocidad as text
% str2double(get(hObject,'String')) returns contents of
velocidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(velocidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.rect.velocidad=velocidad; %Almacenar en puntero handles
pero de velocidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function velocidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function exploradores_Callback(hObject, eventdata, handles)
% hObject handle to exploradores (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of exploradores as
text
% str2double(get(hObject,'String')) returns contents of
exploradores as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
exploradores = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(exploradores)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.rect.exploradores=exploradores; %Almacenar en puntero
handles pero de exploradores
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function exploradores_CreateFcn(hObject, eventdata, handles)
% hObject handle to exploradores (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject handle to tablas (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
open ('C:\MATLAB7\work\tablas.pdf');
```

```
function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of z as text
% str2double(get(hObject,'String')) returns contents of z as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
```

```

z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject,'String',0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.rect.z=z; %Almacenar en puntero handles pero de z
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in simulacion.
function simulacion_Callback(hObject, eventdata, handles)
% hObject handle to simulacion (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
winopen('rectangular_independiente.exe');
pause
winopen('rectangular_completa.exe');
pause
winopen('rectangula_girosbusqueda.exe');

function c_Callback(hObject, eventdata, handles)
% hObject handle to c (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of c as text
% str2double(get(hObject,'String')) returns contents of c as a
double

% --- Executes during object creation, after setting all properties.
function c_CreateFcn(hObject, eventdata, handles)
% hObject handle to c (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
end

```

```

function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of cubo as text
% str2double(get(hObject,'String')) returns contents of cubo as
a double
%-----calculo por ley del cubo-----
Pd=handles.rect.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

%-----FIN-----

% --- Executes during object creation, after setting all properties.
function cubo_CreateFcn(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
end

```

```

% --- Executes on button press in limpiar.
function limpiar_Callback(hObject, eventdata, handles)
% hObject    handle to limpiar (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

initialize_gui(gcbof, handles, true);

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are we are just re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles,'exp') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.rect.anchobarrido = 0;
handles.rect.latitud = 0;
handles.rect.velocidad = 0;
handles.rect.probabilidad = 0;
handles.rect.altitud = 0;
handles.rect.exploradores = 0;
handles.rect.z = 0;
handles.rect.valor = 0;

set(handles.anchobarrido,'String',handles.rect.anchobarrido);
set(handles.latitud,'String',handles.rect.latitud);
set(handles.velocidad,'String',handles.rect.velocidad);
set(handles.probabilidad,'String',handles.rect.probabilidad);
set(handles.altitud,'String',handles.rect.altitud);
set(handles.exploradores,'String',handles.rect.exploradores);
set(handles.z,'String',handles.rect.z);
set(handles.valor,'String',handles.rect.valor);

set(handles.derrota,'String',ini);
set(handles.simultaneo,'String',ini);
set(handles.distsimultaneos,'String',ini);
set(handles.disbusqueda,'String',ini);
set(handles.hrsimultaneo,'String',ini);
set(handles.hrsbusqueda,'String',ini);
set(handles.busqueda,'String',ini);

```

Pantalla: búsqueda de expansion cuadrada:

```

function varargout = expansion(varargin)
% EXPANSION M-file for expansion.fig
%     EXPANSION, by itself, creates a new EXPANSION or
raises the existing
%     singleton*.
%
%     H = EXPANSION returns the handle to a new EXPANSION
or the handle to
%     the existing singleton*.
%
%     EXPANSION('CALLBACK',hObject,eventData,handles,...)
calls the local

```

```

%     function named CALLBACK in EXPANSION.M with the
given input arguments.
%
%     EXPANSION('Property','Value',...) creates a new
EXPANSION or raises the
%     existing singleton*. Starting from the left, property value
pairs are
%     applied to the GUI before expansion_OpeningFunction gets
called. An
%     unrecognized property name or invalid value makes property
application
%     stop. All inputs are passed to expansion_OpeningFcn via
varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

```

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help expansion

% Last Modified by GUIDE v2.5 19-May-2007 23:44:59

```

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',    mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @expansion_OpeningFcn, ...
    'gui_OutputFcn', @expansion_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

```

```

% --- Executes just before expansion is made visible.
function expansion_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin    command line arguments to expansion (see
VARARGIN)

```

```

% Choose default command line output for expansion
handles.output = hObject;

```

```

% Update handles structure
guidata(hObject, handles);

```

```

initialize_gui(hObject, handles, false);

```

```

% UIWAIT makes expansion wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

```

% --- Outputs from this function are returned to the command line.

```
function varargout = expansion_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
%Colocar Imagen de piedra
piedra = imread('C:\MATLAB7\work\piedras.jpg'); %Leer imagen
axes(handles.piedra); %Carga la imagen en fondo
axis off;
imshow(piedra); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

```
%titulos para los cuadros de texto
text(115,15,'Anota los siguientes datos','Fontname','Arial','FontSize',12,'Fontangle','Italic','Fontweight','Bold','color',[1 1 1]);
text(25,35,'Ancho de barrido (millas)','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(129,35,'Area de busqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(195,50,'Latitud','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(195,75,'Altitud','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(260,35,'Velocidad (Nudos)','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(30,67,'Probabilidad [ 0 - 100 ] %','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(30,97,'Según tablas de la normal','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(30,118,'con una probabilidad del:','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(69,127,'%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(175,97,'Si desea usar el [C]','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(175,103,'obtenido,anotelo:','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(175,109,'en caso contrario','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(175,115,'deme su valor de C:','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(60,210,'Derrota ( S )','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(115,220,'Millas Nauticas','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
```

```
text(205,210,'Distancia Recorrida','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(112,260,'Tiempo que Dura la Búsqueda','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(195,280,'Hrs','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
```

```
function anchobarrido_Callback(hObject, eventdata, handles)
% hObject handle to anchobarrido (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of anchobarrido as text
% str2double(get(hObject,'String')) returns contents of anchobarrido as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
anchobarrido = str2double(NewStrVal); %Transformar a formato double o de string a numerico
if isnan(anchobarrido)
    set(hObject,'String',0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.exp.anchobarrido=anchobarrido; %Almacenar en puntero handles pero del anchobarrido
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function anchobarrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to anchobarrido (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
```

```
function latitud_Callback(hObject, eventdata, handles)
% hObject handle to latitud (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of latitud as text
% str2double(get(hObject,'String')) returns contents of latitud as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
latitud = str2double(NewStrVal); %Transformar a formato double o de string a numerico
if isnan(latitud)
    set(hObject,'String',0);
    errordlg('Solo debes introducir numeros','Error');
end
```

```
handles.exp.latitud=latitud; %Almacenar en puntero handles pero
de latitud
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function latitud_CreateFcn(hObject, eventdata, handles)
% hObject handle to latitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function altitud_Callback(hObject, eventdata, handles)
% hObject handle to altitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of altitud as text
% str2double(get(hObject,'String')) returns contents of altitud
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
altitud = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(altitud)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.exp.altitud=altitud; %Almacenar en puntero handles pero
de altitud
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function altitud_CreateFcn(hObject, eventdata, handles)
% hObject handle to altitud (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
```

```
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad=str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.exp.probabilidad=probabilidad; %Almacenar en puntero
handles pero de busqueda
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
%----- VARIABLES PARA DETERMINAR DERROTAS (S) -----
Z=handles.exp.z;
W=handles.exp.anchobarrido;
Sd= int8(1.253*(W/Z));
set(handles.derrota,'String',Sd);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN -----
```

```
%-----VARIABLES PARA DETERMINAR DISTANCIA-----
if handles.exp.altitud==handles.exp.latitud
    a=handles.exp.altitud;
    S = char(Sd)
    p = int8(a/S)
    x= char(p)- 1;
    Di = 0;
    for k = 1:1:x-1
        Di = Di + (2*(k*S));
    end
    Df = Di + 3*(x*S);
    set(handles.distancia,'String',Df);
```

```

else
    if handles.exp.altitud>handles.exp.latitud
        LM=handles.exp.altitud;
        LMEN=handles.exp.latitud;
    else
        LM=handles.exp.latitud;
        LMEN=handles.exp.altitud;
    end
    x = (LM/S) - 1;
    y = (LMEN/S) - 1;
    Di = 0;
    for k = 1:1:y-1
        Di = Di + (2*(k*S));
    end
    Df = Di + (3*(y*S)) + (x*S);
    set(handles.distancia,'string',Df);

end
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata

%-----FIN-----

%-----VARIABLES PARA DETERMINAR TIEMPO DE
BUSQUEDA -----
V=handles.exp.velocidad;
TIEMPO2 = Df/V ;
tiempo3 = time2str(TIEMPO2)
set(handles.tiempo,'String',tiempo3);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

function velocidad_Callback(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of velocidad as text
% str2double(get(hObject,'String')) returns contents of
velocidad as a double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(velocidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.exp.velocidad=velocidad; %Almacenar en puntero handles
pero de velocidad
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function velocidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in simulacion.
function simulacion_Callback(hObject, eventdata, handles)
% hObject handle to simulacion (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
winopen('expansion_texto.exe');

function c_Callback(hObject, eventdata, handles)
% hObject handle to c (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of c as text

```

```

%      str2double(get(hObject,'String')) returns contents of c as a
double

% --- Executes during object creation, after setting all properties.
function c_CreateFcn(hObject, eventdata, handles)
% hObject    handle to c (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
%      See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject    handle to cubo (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
%-----calculo por ley del cubo-----
Pd=handles.exp.probabilidad;
P = Pd/100;
tabla = P/2 + 0.5;
C=norminv(tabla, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

function valor_Callback(hObject, eventdata, handles)
% hObject    handle to valor (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of valor as text
%      str2double(get(hObject,'String')) returns contents of valor as
a double

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to valor (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
%      See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in limpiar.

```

```

function limpiar_Callback(hObject, eventdata, handles)
% hObject    handle to limpiar (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

initialize_gui(gcbf, handles, true);

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are just re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles,'exp') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.exp.anchobarrido = 0;
handles.exp.latitud = 0;
handles.exp.velocidad = 0;
handles.exp.probabilidad = 0;
handles.exp.altitud = 0;
handles.exp.z = 0;

set(handles.anchobarrido,'String',handles.exp.anchobarrido);
set(handles.latitud,'String',handles.exp.latitud);
set(handles.velocidad,'String',handles.exp.velocidad);
set(handles.probabilidad,'String',handles.exp.probabilidad);
set(handles.altitud,'String',handles.exp.altitud);
set(handles.z,'String',handles.exp.z);

set(handles.derrota,'String',ini);
set(handles.distancia,'String',ini);
set(handles.tiempo,'String',ini);
set(handles.valor,'String',ini);

```

Pantalla de búsqueda por sector en base fija:

```

function varargout = sector_base_fija(varargin)
% SECTOR_BASE_FIJA M-file for sector_base_fija.fig
%      SECTOR_BASE_FIJA, by itself, creates a new
SECTOR_BASE_FIJA or raises the existing
%      singleton*.
%
%      H = SECTOR_BASE_FIJA returns the handle to a new
SECTOR_BASE_FIJA or the handle to
%      the existing singleton*.
%
%      SECTOR_BASE_FIJA('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in SECTOR_BASE_FIJA.M
with the given input arguments.
%
%      SECTOR_BASE_FIJA('Property','Value',...) creates a new
SECTOR_BASE_FIJA or raises the
%      existing singleton*. Starting from the left, property value
pairs are
%      applied to the GUI before sector_base_fija_OpeningFunction
gets called. An
%      unrecognized property name or invalid value makes property
application
%      stop. All inputs are passed to sector_base_fija_OpeningFcn
via varargin.

```

```

%
%      *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help
sector_base_fija

% Last Modified by GUIDE v2.5 14-Jul-2007 07:24:55

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @sector_base_fija_OpeningFcn, ...
    'gui_OutputFcn',  @sector_base_fija_OutputFcn, ...
    'gui_LayoutFcn', [] , ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before sector_base_fija is made visible.
function sector_base_fija_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to sector_base_fija (see
VARARGIN)

% Choose default command line output for sector_base_fija
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

initialize_gui(hObject, handles, false);

% UIWAIT makes sector_base_fija wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = sector_base_fija_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

%Colocar Imagen de piedra

```

```

fondo = imread('C:\MATLAB7\work\pedras2.jpg'); %Leer imagen
axes(handles.fondo); %Carga la imagen en fondo
axis off;
imshow(fondo); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

%Colocar Imagen de fondo
%Carga la imagen de fondo (opcional)
%[x,map]=imread('pedras2.jpg','jpg');
%image(x),colormap(map),axis off,hold on

%titulos para los cuadros de texto
text(205,30,'Anota los siguientes
datos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(25,63,'Radio de busqueda
(millas)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(315,63,'Sector delimitado entre
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(385,88,'y','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fon
tweight','Normal','color',[1 1 1]);
text(230,91,'MV
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(525,91,'MV','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(30,130,'Probabilidad [ 0 - 100 ]
%','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','No
rmal','color',[1 1 1]);
text(242,130,'Ancho de barrido (Millas)
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(464,130,'Velocidad
(Nudos)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(30,191,'Según tablas de la normal
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(30,230,'con una probabilidad del:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(110,245,'%','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(290,191,'Si desea usar el
[C]','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(290,204,'obtenido,anotelo:','Fontname','Arial','FontSize',9,'Font
angle','Italic','Fontweight','Normal','color',[1 1 1]);
text(290,216,'en caso contrario
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(290,228,'deme su valor de C:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(230,320,'Resultados
Teóricos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontwe
ight','Bold','color',[1 1 1]);
text(10,360,'Derrota ( S ) Corregida ( Sc
)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(216,360,' Long. del arco N° Derrotas
N° de
exploradores','Fontname','Arial','FontSize',9,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(325,383,'MN','Fontname','Arial','FontSize',9,'Fontangle','Italic','
Fontweight','Normal','color',[1 1 1]);

```

```

text(10,430,'Long. del subsector          Angulo del subsector
( S / 2 )°          Nva. Prob.          (Cc) Corregido
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(119,455,'MN                                grados
(%)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(200,490,'Rumbo          de          los          Exploradores
','Fontname','Arial','FontSize',12,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(15,515,'Explorador
Nº','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(168,515,'R.Salida','Fontname','Arial','FontSize',9,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(345,515,'R.Regreso','Fontname','Arial','FontSize',9,'Fontangle','
Italic','Fontweight','Normal','color',[1 1 1]);
text(510,515,'R.Transversal','Fontname','Arial','FontSize',9,'Fontang
le','Italic','Fontweight','Normal','color',[1 1 1]);

```

```

function radio_Callback(hObject, eventdata, handles)
% hObject    handle to radio (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of radio as text
%        str2double(get(hObject,'String')) returns contents of radio as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
radio = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(radio)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.radio=radio; %Almacenar en puntero handles pero del
radio
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function radio_CreateFcn(hObject, eventdata, handles)
% hObject    handle to radio (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
%        See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function mvf_Callback(hObject, eventdata, handles)
% hObject    handle to mvf (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of mvf as text
%        str2double(get(hObject,'String')) returns contents of mvf as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
mvf = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(mvf)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.mvf=mvf; %Almacenar en puntero handles pero del
mvf
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function mvf_CreateFcn(hObject, eventdata, handles)
% hObject    handle to mvf (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
%        See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function mvi_Callback(hObject, eventdata, handles)
% hObject    handle to mvi (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of mvi as text
%        str2double(get(hObject,'String')) returns contents of mvi as a
double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
mvi = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(mvi)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.mvi=mvi; %Almacenar en puntero handles pero del
mvi
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function mvi_CreateFcn(hObject, eventdata, handles)
% hObject    handle to mvi (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
%        See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));

```

end

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.probabilidad=probabilidad; %Almacenar en puntero
handles pero del probabilidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function velocidad_Callback(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velocidad as text
% str2double(get(hObject,'String')) returns contents of
velocidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(velocidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velocidad=velocidad; %Almacenar en puntero
handles pero del velocidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function velocidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function barrido_Callback(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of barrido as text
% str2double(get(hObject,'String')) returns contents of barrido
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
barrido = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(barrido)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.barrido=barrido; %Almacenar en puntero handles
pero del barrido
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function barrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of z as text
```

```

%      str2double(get(hObject,'String')) returns contents of z as a
double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.z=z; %Almacenar en puntero handles pero del c
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject    handle to z (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
%      See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject    handle to tablas (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

open('C:\MATLAB7\work\tablas.pdf');

% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject    handle to buscar (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

%----- VARIABLES PARA DETERMINAR DERROTAS (S) La
cual no aparecerá en pantalla-----
Z=handles.base.z;
W=handles.base.barrido;
S=1.253*(W/Z);
set(handles.s,'string',S);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----LONGITUD DEL ARCO-----
if handles.base.mvi<handles.base.mvf
    mv = handles.base.mvf - handles.base.mvi;
else
    mv = handles.base.mvi - handles.base.mvf;
end
R = handles.base.radio;
arco = ( mv * 6.28 * R ) / 360;
set(handles.arco,'String',arco);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----NUMERO DE DERROTAS -----

```

```

Di = int8((arco) / (S));% int8 es para redondear el resultado al
entero mas proximo
x= rem(Di,2);% me muestra el residuo entre la division de di y 2

```

```

if x == 0
    set(handles.nderrota,'String',Di);% me muestra Di en nderrota
else
    Di = Di + 1;
    x = rem (Di,2);
    if x == 0
        set(handles.nderrota,'String',Di);
    else
        end
end

```

```

guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%--DEBIDO AL CALCULO NUEVO EN NUMERO DE
DERROTAS SE VUELVE A CALCULAR S
%LLAMANDOLA Sc Y COMO C ESTA RELACIONADO CON
S SE RECALCULA LLAMANDOLA Cc--
Df = char(Di); %Di se convierte de nuevo a string ya que mantenía
un resultado redondeado y en Sc el resultado saldria redondeado
Sc = arco/Df;
Cc = W/Sc ;
set(handles.derrota,'String',Sc);
set(handles.cc,'String',Cc);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----CALCULO DEL NUMERO DE EXPLORADORES-----
-----

```

```

N = Df/2;
set(handles.exploradores,'String',N);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----LONGITUD DEL SUBSECTOR-----
-----

```

```

subsector= arco/N;
set(handles.subsector,'String',subsector);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----ANGULO DEL SUBSECTOR-----
-----

```

```

angulo = mv/N;
set(handles.angulo,'String',angulo);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----CALCULO DE S/2-----
-----

```

```

mediaS = ((Sc/2)* angulo)/subsector;
set(handles.mitads,'String',mediaS);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----colocando rumbos-----
-----

```

```

if handles.base.mvi<handles.base.mvf
    chico = handles.base.mvi;
else
    chico = handles.base.mvf;
end

```

```

for k = 1:1:N;
    j = 0:1:N-1;
    b = 1:1:N;
    h = 1:1:N;

```

```

Rksal= (chico + j*angulo) + mediaS;
Rkreg= (chico + (h*angulo) + 180) - mediaS;
Rktran= Rksal + 90 + mediaS;

set(handles.salida,'String',Rksal);
set(handles.regreso,'String',Rkreg);
set(handles.transversal,'String',Rktran);
set(handles.explorador,'String',h);
end
guidata(hObject,handles);
%-----FIN-----

%-----NUEVA PROBABIIDAD-----
%-----
nueva=normcdf(Cc,0,1);
probanueva= ((nueva - 0.5)*2)*100;
set(handles.probn,'String',probanueva);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

Pd=handles.base.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in limpiar.
function limpiar_Callback(hObject, eventdata, handles)
% hObject handle to limpiar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

initialize_gui(gcf, handles, true);

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are we are just re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles,'base') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.base.radio = 0;
handles.base.mvi = 0;
handles.base.mvf = 0;
handles.base.probabilidad = 0;
handles.base.barrido = 0;
handles.base.velocidad = 0;
handles.base.z = 0;

set(handles.radio,'String',handles.base.radio);
set(handles.mvf,'String',handles.base.mvf);
set(handles.mvi,'String',handles.base.mvi);
set(handles.probabilidad,'String',handles.base.probabilidad);
set(handles.velocidad,'String',handles.base.velocidad);
set(handles.barrido,'String',handles.base.barrido);
set(handles.z,'String',handles.base.z);

set(handles.derrota,'String',ini);
set(handles.nderrota,'String',ini);
set(handles.arco,'String',ini);
set(handles.exploradores,'String',ini);
set(handles.valor,'String',ini);
set(handles.subsector,'String',ini);
set(handles.angulo,'String',ini);
set(handles.mitads,'String',ini);
set(handles.explorador,'String',ini);
set(handles.salida,'String',ini);
set(handles.transversal,'String',ini);
set(handles.regreso,'String',ini);
set(handles.s,'String',ini);
set(handles.cc,'String',ini);
set(handles.probn,'String',ini);

% --- Executes on button press in simular.
function simular_Callback(hObject, eventdata, handles)
% hObject handle to simular (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
winopen('sector_base_fija_texto.exe');

```

Pantalla de búsqueda de sector en base móvil:

```

function varargout = sector_base_movil(varargin)
% SECTOR_BASE_MOVIL M-file for sector_base_movil.fig
% SECTOR_BASE_MOVIL, by itself, creates a new
SECTOR_BASE_MOVIL or raises the existing
% singleton*.
%
% H = SECTOR_BASE_MOVIL returns the handle to a new
SECTOR_BASE_MOVIL or the handle to
% the existing singleton*.
%
%
SECTOR_BASE_MOVIL('CALLBACK',hObject,eventData,handl
es,...) calls the local
% function named CALLBACK in
SECTOR_BASE_MOVIL.M with the given input arguments.
%
% SECTOR_BASE_MOVIL('Property','Value',...) creates a new
SECTOR_BASE_MOVIL or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before
sector_base_movil_OpeningFunction gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to
sector_base_movil_OpeningFcn via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
% instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help
sector_base_movil

% Last Modified by GUIDE v2.5 19-Nov-2007 11:34:08

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
'gui_Singleton', gui_Singleton, ...
'gui_OpeningFcn', @sector_base_movil_OpeningFcn,
...
'gui_OutputFcn', @sector_base_movil_OutputFcn, ...
'gui_LayoutFcn', [] , ...
'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before sector_base_movil is made visible.
function sector_base_movil_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% varargin command line arguments to sector_base_movil (see
VARARGIN)

```

```

% Choose default command line output for sector_base_movil
handles.output = hObject;

```

```

% Update handles structure
guidata(hObject, handles);

```

```

initialize_gui(hObject, handles, false);

```

```

% UIWAIT makes sector_base_movil wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

```

```

% --- Outputs from this function are returned to the command line.
function varargout = sector_base_movil_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Get default command line output from handles structure
varargout{1} = handles.output;

```

```

%Colocar Imagen de piedra
%sector_base_movil = imread('C:\MATLAB7\work\piedras2.jpg');
%Leer imagen
%axes(handles.sector_base_movil); %Carga la imagen en
sector_base_movil
%axis off;
%imshow(sector_base_movil); %Presenta la imagen
%handles.output = hObject;
% Update handles structure
%guidata(hObject, handles);
[x,map]=imread('piedras2.jpg','jpg');
image(x),colormap(map),axis off,hold on

```

```

%Colocar Imagen de sector_base_movil
%Carga la imagen de sector_base_movil (opcional)
%[x,map]=imread('piedras2.jpg','jpg');
%image(x),colormap(map),axis off,hold on

```

```

%titulos para los cuadros de texto
text(205,30,'Anota los siguientes
datos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(20,63,'Radio de
busqueda','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(110,88,'millas','Fontname','Arial','FontSize',9,'Fontangle','Italic'
,'Fontweight','Normal','color',[1 1 1]);
text(156,63,'Vel. del portaviones
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(289,63,'Rumbo del portaviones
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(425,63,'Sector de
búsqueda','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(247,88,'Kts','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(382,88,'grados
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);

```

```

text(515,88,'grados','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(30,128,'Probabilidad [ 0 - 100 ]',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(205,128,'Ancho de barrido (MN)',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(360,128,'Velocidad de exploradores (Kts)',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(30,191,'Según tablas de la normal',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(30,230,'con una probabilidad del:',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(110,245,'%','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight',
'Normal','color',[1 1 1]);
text(240,191,'Si desea usar el [C]',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(240,204,'obtenido,anotelo:',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(240,216,'en caso contrario',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(240,228,'deme su valor de C:',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(230,320,'Resultados Teóricos','Fontname','Arial','FontSize',13,'Fontangle','Italic',
'Fontweight','Bold','color',[1 1 1]);
text(36,358,'Derrota (S)','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(128,358,'Long. de arco','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(193,383,'MN','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(228,358,'Nº Derrotas','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(313,358,'Nº de exploradores','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(417,358,'Long. del subsector','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(492,383,'MN','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(535,358,'Angulo del subsector','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(599,383,'grados','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(18,430,'Calculo de Distancias (M.N.)','Fontname','Arial','FontSize',9,
'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(15,490,'1era.Pierna','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(99,490,'2da.Pierna','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(10,530,'Datos para calcular 3era.Pierna','Fontname','Arial','FontSize',9,
'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(10,580,'Dist. Recorrida','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);

```

```

text(10,594,'al final de pierna','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(20,607,'transversal','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(102,580,'Tiempo (Hrs)','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(181,430,'Rumbo de los Exploradores (El R.regreso se obtiene graficamente)',
'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal',
'color',[1 1 1]);
text(200,450,'Explorador Nº','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(289,450,'R.Salida','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(389,450,'R.Transversal','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(220,643,'Rumbo en límite derecho del subsector respectivo','Fontname',
'Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(503,487,'El Rumbo de Regreso','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(503,507,'se determina graficamente','Fontname','Arial','FontSize',9,
'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(507,550,'Derrota','Fontname','Arial','FontSize',9,'Fontangle','Italic',
'Fontweight','Normal','color',[1 1 1]);
text(497,563,'corregida (Sc)','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(577,550,'Factor de Barrido','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(581,563,'corregido (Cc)','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);
text(11,643,'Al termino de segunda pierna','Fontname','Arial','FontSize',9,
'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(11,657,'el portaviones habrá recorrido','Fontname','Arial','FontSize',9,
'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(11,669,'una distancia de','Fontname','Arial','FontSize',9,'Fontangle',
'Italic','Fontweight','Normal','color',[1 1 1]);

```

```

function radio_Callback(hObject,eventdata,handles)
% hObject handle to radio (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of radio as text
% str2double(get(hObject,'String')) returns contents of radio as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
radio = str2double(NewStrVal); %Transformar a formato double o de string a numerico
if isnan(radio)
    set(hObject,'String',0);
    error('Solo debes introducir numeros','Error');
end
handles.base.radio=radio; %Almacenar en puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function radio_CreateFcn(hObject,eventdata,handles)

```

```
% hObject handle to radio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function rumboporta_Callback(hObject, eventdata, handles)
% hObject handle to rumboporta (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of rumboporta as
text
```

```
% str2double(get(hObject,'String')) returns contents of
rumboporta as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
rumboporta = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
```

```
if isnan(rumboporta)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.rumboporta=rumboporta; %Almacenar en puntero
handles pero del mvf
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function rumboporta_CreateFcn(hObject, eventdata, handles)
% hObject handle to rumboporta (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function velporta_Callback(hObject, eventdata, handles)
% hObject handle to velporta (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velporta as text
% str2double(get(hObject,'String')) returns contents of velporta
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
```

```
velporta = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
```

```
if isnan(velporta)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velporta=velporta; %Almacenar en puntero handles
pero del mvi
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function velporta_CreateFcn(hObject, eventdata, handles)
% hObject handle to velporta (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
```

```
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
```

```
if isnan(probabilidad)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.probabilidad=probabilidad; %Almacenar en puntero
handles pero del probabilidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function velexplorador_Callback(hObject, eventdata, handles)
% hObject handle to velexplorador (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velexplorador as
text
% str2double(get(hObject,'String')) returns contents of
velexplorador as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velexplorador = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(velexplorador)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velexplorador=velexplorador; %Almacenar en
puntero handles pero del velocidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function velexplorador_CreateFcn(hObject, eventdata, handles)
% hObject handle to velexplorador (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function barrido_Callback(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of barrido as text
% str2double(get(hObject,'String')) returns contents of barrido
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
barrido = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(barrido)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.barrido=barrido; %Almacenar en puntero handles
pero del barrido
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function barrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of z as text
% str2double(get(hObject,'String')) returns contents of z as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.z=z; %Almacenar en puntero handles pero del c
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject handle to tablas (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
open('C:\MATLAB7\work\tablas.pdf');
```

```
% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
```

```

% handles structure with handles and user data (see GUIDATA)

%----- VARIABLES PARA DETERMINAR DERROTAS (S) La
cual no aparecerá en pantalla-----
Z=handles.base.z;
W=handles.base.barrido;
S=1.253*(W/Z);
set(handles.derrota,'String',S);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----LONGITUD DEL ARCO-----
angulosector= handles.base.sector;
R = handles.base.radio;
arco = ( angulosector * 2 * pi * R ) / 360;
set(handles.arco,'String',arco);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----NUMERO DE DERROTAS -----
Di = int8((arco) / (S));% int8 es para redondear el resultado al
entero mas proximo
x= rem(Di,2);% me muestra el residuo entre la division de di y 2

if x == 0
set(handles.nderrota,'String',Di);% me muestra Di en nderrota
set(handles.maxmin,'string','prob. de contacto minimo');
else
Di = Di + 1;
x = rem (Di,2);
if x == 0
set(handles.nderrota,'String',Di);
set(handles.maxmin,'string','prob. de contacto minimo');
else
end
end

guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%--DEBIDO AL CALCULO NUEVO EN NUMERO DE
DERROTAS SE VUELVE A CALCULAR S
%LLAMANDOLA Sc Y COMO C ESTA RELACIONADO CON
S SE RECALCULA LLAMANDOLA Cc--
Df = char(Di); %Di se convierte de nuevo a string ya que mantenía
un resultado redondeado y en Sc el resultado saldría redondeado
Sc = arco/Df;
Cc = W/Sc ;
set(handles.sc,'String',Sc);
set(handles.cc,'String',Cc);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----CALCULO DEL NUMERO DE EXPLORADORES-----
-----
N = Df/2;
set(handles.exploradores,'String',N);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----LONGITUD DEL SUBSECTOR-----
-----
subsector= arco/N;
set(handles.subsector,'String',subsector);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----ANGULO DEL SUBSECTOR-----
-----
angulo = angulosector/N;
set(handles.angulo,'String',angulo);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata

```

```

%-----FIN-----

%-----CALCULO DE DISTANCIAS-----
-----
piernauno= R - ( Sc / 2 );
set(handles.piernauno,'String',piernauno);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata

piernados = Sc;
set(handles.piernados,'String',piernados);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata

%--datos 3era pierna-----

DF = piernauno + Sc;
set(handles.DF,'String',DF);

velexp = handles.base.velexplorador;
tiempo = DF / velexp;
set(handles.tiempo,'String',tiempo);
%-----FIN-----

%-----CALCULO DE S/2-----
-----
mediaS = ((Sc/2)* angulo)/subsector;
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----colocando rumbos-----
if handles.base.sector>handles.base.rumboporta
angmayor = handles.base.sector;
angmenor = handles.base.rumboporta;
chico = angmayor - angmenor;
else
angmayor = handles.base.rumboporta;
angmenor = handles.base.sector;
chico = angmayor - angmenor;
end

for k = 1:1:N-1;
j = 0:1:N-2 ;

h = 1:1:N-1;

Rksal= (chico + j*angulo) + mediaS;

Rktran= Rksal + 90 + mediaS;

set(handles.salida,'String',Rksal);
set(handles.transversal,'String',Rktran);
set(handles.explorador,'String',h);

end

k = N;
%j = N-1;
%g = N-2;
h = N;

Rksaln= (angmayor + chico) - mediaS;

Rktrann = Rksaln-90 - mediaS + 360;

```

```

set(handles.saln,'String',Rksaln);
set(handles.transn,'String',Rktrann);
set(handles.expln,'String',h);

guidata(hObject,handles);
%-----FIN-----

%---DISTANCIA RECORRIDA POR PORTAVIONES AL
TERMINO DE SEGUNDA PIERNA-----
V = handles.base.velporta;
distporta= V * tiempo;
set(handles.distporta,'String',distporta);
guidata(hObject,handles);
%-----FIN-----

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

Pd=handles.base.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in limpiar.
function limpiar_Callback(hObject, eventdata, handles)
% hObject handle to limpiar (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

initialize_gui(gcbf, handles, true);

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are we are just re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles,'base') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.base.radio = 0;
handles.base.mvi = 0;
handles.base.mvf = 0;
handles.base.probabilidad = 0;
handles.base.barrido = 0;
handles.base.velocidad = 0;
handles.base.z = 0;
handles.base.sector = 0;

set(handles.sector,'String',handles.base.sector);
set(handles.radio,'String',handles.base.radio);
set(handles.rumboporta,'String',handles.base.mvf);
set(handles.velporta,'String',handles.base.mvi);
set(handles.probabilidad,'String',handles.base.probabilidad);
set(handles.velexplorador,'String',handles.base.velocidad);
set(handles.barrido,'String',handles.base.barrido);
set(handles.z,'String',handles.base.z);

set(handles.derrota,'String',ini);
set(handles.nderrota,'String',ini);
set(handles.arco,'String',ini);
set(handles.exploradores,'String',ini);
set(handles.valor,'String',ini);
set(handles.subsector,'String',ini);
set(handles.angulo,'String',ini);
set(handles.piernauno,'String',ini);
set(handles.piernados,'String',ini);
set(handles.explorador,'String',ini);
set(handles.salida,'String',ini);
set(handles.transversal,'String',ini);
set(handles.saln,'String',ini);
set(handles.DF,'String',ini);
set(handles.tiempo,'String',ini);
set(handles.expln,'String',ini);
set(handles.transn,'String',ini);
set(handles.sc,'String',ini);
set(handles.cc,'String',ini);
set(handles.distporta,'String',ini);

function sector_Callback(hObject, eventdata, handles)
% hObject handle to sector (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sector as text
% str2double(get(hObject,'String')) returns contents of sector
as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado

```

```

sector = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(sector)
    set(hObject,'String', 0);
    errorlg('Solo debes introducir numeros','Error');
end
handles.base.sector=sector; %Almacenar en puntero handles pero
del mvf
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function sector_CreateFcn(hObject, eventdata, handles)
% hObject handle to sector (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in similar.
function similar_Callback(hObject, eventdata, handles)
% hObject handle to similar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
winopen('sector_base_movil.exe');

```

Pantalla de búsqueda por delante:

```

function varargout = busqueda_delante(varargin)
% BUSQUEDA_DELANTE M-file for busqueda_delante.fig
% BUSQUEDA_DELANTE, by itself, creates a new
BUSQUEDA_DELANTE or raises the existing
% singleton*.
%
% H = BUSQUEDA_DELANTE returns the handle to a new
BUSQUEDA_DELANTE or the handle to
% the existing singleton*.
%
%
BUSQUEDA_DELANTE('CALLBACK',hObject,eventData,handl
es,...) calls the local
% function named CALLBACK in
BUSQUEDA_DELANTE.M with the given input arguments.
%
% BUSQUEDA_DELANTE('Property','Value',...) creates a new
BUSQUEDA_DELANTE or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before
busqueda_delante_OpeningFunction gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to busqueda_delante_OpeningFcn
via varargin.
%

```

```

% *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

```

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help
busqueda_delante

% Last Modified by GUIDE v2.5 13-Oct-2007 21:29:21

```

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @busqueda_delante_OpeningFcn, ...
    'gui_OutputFcn', @busqueda_delante_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

```

```

% --- Executes just before busqueda_delante is made visible.
function busqueda_delante_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to busqueda_delante (see
VARARGIN)

```

% Choose default command line output for busqueda_delante
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

initialize_gui(hObject, handles, false);

% UIWAIT makes busqueda_delante wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

```

% --- Outputs from this function are returned to the command line.
function varargout = busqueda_delante_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

% Get default command line output from handles structure
varargout{1} = handles.output;

%Colocar Imagen de piedra
fondo = imread('C:\MATLAB7\work\pedras2.jpg'); %Leer imagen

```

axes(handles.fondo); %Carga la imagen en fondo
axis off;
imshow(fondo); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

%Colocar Imagen de fondo
%Carga la imagen de fondo (opcional)
%[x,map]=imread('piedras2.jpg','jpg');
%image(x),colormap(map),axis off,hold on

%titulos para los cuadros de texto
text(205,20,'Anota los siguientes
datos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(85,59,'Buque
Propio','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweig
ht','Bold','color',[1 1 1]);
text(415,59,'Buque a localizar
','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweight','bol
d','color',[1 1 1]);
text(25,83,'Velocidad','Fontname','Arial','FontSize',9,'Fontangle','Ita
lic','Fontweight','Normal','color',[1 1 1]);
text(93,106,'Nudos','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(372,83,'Reporte','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(370,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(459,83,'Zarpar','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(458,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(557,93,'MV','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(605,116,'grados','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(146,83,'Probabilidad
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(226,106,[' 0 - 100 ]
%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','No
rmal','color',[1 1 1]);
text(25,137,'Ancho de barrido
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(95,162,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(370,147,'Distancia','Fontname','Arial','FontSize',9,'Fontangle','I
talic','Fontweight','Normal','color',[1 1 1]);
text(437,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(489,147,'Error de
posición','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(555,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(430,200,'Velocidades
(Nudos)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(387,214,'Vmin','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(473,214,'Vprob','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(562,214,'Vmax','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(149,137,'Según tablas de la normal
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);

```

```

text(149,179,'con una probabilidad del:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(216,194,'%','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(25,221,'Si desea usar el
[C]','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(25,234,'obtenido,anotelo:','Fontname','Arial','FontSize',9,'Fonta
ngle','Italic','Fontweight','Normal','color',[1 1 1]);
text(25,246,'en caso contrario
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(25,258,'deme su valor de C:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(372,273,'Frente de Busqueda F
=','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','No
rmal','color',[1 1 1]);
text(584,276,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(372,309,'Tiempo de intercepción
(hrs)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(230,410,'Resultados
Teóricos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontwe
ight','Bold','color',[1 1 1]);
text(55,440,'RV
(
grados)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(263,440,'Hora de int.
(Hrs.)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(488,440,'Nº de
exploradores','Fontname','Arial','FontSize',9,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(55,500,'Derrota (S)
Corregida
(Sc)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(140,520,'Millas
Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight
','Normal','color',[1 1 1]);
text(448,500,'Nva. Prob.
Contacto','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(481,520,'( 0 - 100
)%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(45,570,'Tiempo Total
(Hrs)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(263,570,'Avance Max
(Millas)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(488,570,'Avance Min
(Millas)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(153,630,'Avance al
principio','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(153,644,'de intercepción Factor
Corregido
(Cc)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(257,672,'Millas
Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight
','Normal','color',[1 1 1]);

```

```
function velocidadpropio_Callback(hObject, eventdata, handles)
```

```
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velocidadpropio
as text
% str2double(get(hObject,'String')) returns contents of
velocidadpropio as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidadpropio = str2double(NewStrVal); %Transformar a
formato double o de string a numerico
if isnan(velocidadpropio)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velocidadpropio=velocidadpropio; %Almacenar en
puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function velocidadpropio_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function distancia_Callback(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of distancia as text
% str2double(get(hObject,'String')) returns contents of
distancia as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
distancia = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(distancia)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.distancia=distancia; %Almacenar en puntero handles
pero del mvf
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function distancia_CreateFcn(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function mvi_Callback(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of mvi as text
% str2double(get(hObject,'String')) returns contents of mvi as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
mvi = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(mvi)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.mvi=mvi; %Almacenar en puntero handles pero del
mvi
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function mvi_CreateFcn(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject,'String', 0);
```

```

    errordlg('Solo debes introducir numeros','Error');
end
handles.base.probabilidad=probabilidad; %Almacenar en puntero
handles pero del probabilidad
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function ep_Callback(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of ep as text
% str2double(get(hObject,'String')) returns contents of ep as a
double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ep = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ep)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.ep=ep; %Almacenar en puntero handles pero del
velocidad
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function ep_CreateFcn(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function barrido_Callback(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of barrido as text
% str2double(get(hObject,'String')) returns contents of barrido
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
barrido = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(barrido)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.barrido=barrido; %Almacenar en puntero handles
pero del barrido
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function barrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of z as text
% str2double(get(hObject,'String')) returns contents of z as a
double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.z=z; %Almacenar en puntero handles pero del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject handle to tablas (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

open ('C:\MATLAB7\work\tablas.pdf');

% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

%----- VARIABLES PARA DETERMINAR DERROTAS (S) La
cual no aparecerá en pantalla-----
Z=handles.base.z;
W=handles.base.barrido;
S=1.253*(W/Z);
set(handles.derrota,'string',S);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN -----

%----AVANCE DEL ENEMIGO SEGUN HORA DE REPORTE
Y HORA DE ZARPAR-----
if handles.base.reporte<handles.base.zarpar
    tm= hms2hr(handles.base.zarpar)- hms2hr(handles.base.reporte);
else
    tm=
        hms2hr(handles.base.reporte)-
hms2hr(handles.bese.zarpar);
end

avanceinicio = (handles.base.vmax * tm);
set(handles.avanceinicio,'string',avanceinicio);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----TIMEPO TOTAL-----
-----
Tt= tm + handles.base.ti;
set(handles.total,'string',Tt);
guidata(hObject, handles);
%-----FIN-----

%-----AVANCE MAXIMO-----
-----
arcoab = Tt * handles.base.vmax;
set(handles.arcoab,'string',arcoab);
guidata(hObject, handles);
%-----FIN-----

%-----AVANCE MINIMO-----
-----
arcocd= Tt * handles.base.vmin;
set(handles.arcocd,'string',arcocd);
guidata(hObject, handles);
%-----FIN-----

```

```

%-----NUMERO DE EXPLORADORES -----
F = handles.base.frente;
if S == W
    N = ceil(F / S);
    Sc = F/N;
    Cc = W / Sc;
    set(handles.exploradores,'string',N);
    set(handles.sc,'string',Sc);
    set(handles.cc,'string',Cc);
    guidata(hObject, handles);
else
    N = ceil(((F - W) / S) + 1);
    Sc = (F - W) / (N - 1);
    Cc = W / Sc;
    set(handles.exploradores,'string',N);
    set(handles.sc,'string',Sc);
    set(handles.cc,'string',Cc);
    guidata(hObject, handles);
end

guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----NUEVA PROBABIIDAD-----
-----
nueva=normcdf(Cc,0,1);
probanueva= ((nueva - 0.5)*2)*100;
set(handles.nuevapro,'String',probanueva);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----HORA INTERSECCIÓN-----
ti = handles.base.ti;
hora= hms2hr(handles.base.zarpar)+ ti;
horainter = time2str(hora);
set(handles.horainter,'String',horainter);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

Pd=handles.base.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

```

```
% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in limpiar.
```

```
function limpiar_Callback(hObject, eventdata, handles)
```

```
% hObject handle to limpiar (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles structure with handles and user data (see GUIDATA)
```

```
initialize_gui(gcbf, handles, true);
```

```
%-----
```

```
function initialize_gui(fig_handle, handles, isreset)
```

```
% If the metricdata field is present and the reset flag is false, it
means
```

```
% we are we are just re-initializing a GUI by calling it from the
cmd line
```

```
% while it is up. So, bail out as we dont want to reset the data.
```

```
if isfield(handles,'base') && ~isreset
```

```
    return;
```

```
end
```

```
%Limpiar pantalla
```

```
ini=char(' ');
```

```
%-----
```

```
handles.base.radio = 0;
handles.base.mvi = 0;
handles.base.reporte = 0;
handles.base.probabilidad = 0;
handles.base.barrido = 0;
handles.base.velocidad = 0;
handles.base.z = 0;
handles.base.reporte = ('0000');
handles.base.zarpar = ('0000');
handles.base.distancia = 0;
handles.base.vmin = 0;
handles.base.vprob = 0;
handles.base.vmax = 0;
handles.base.frente = 0;
handles.base.ti = 0;
```

```
set(handles.velocidadpropio,'String',handles.base.radio);
set(handles.reporte,'String',handles.base.reporte);
set(handles.zarpar,'String',handles.base.zarpar);
set(handles.mvi,'String',handles.base.mvi);
set(handles.probabilidad,'String',handles.base.probabilidad);
set(handles.ep,'String',handles.base.velocidad);
set(handles.barrido,'String',handles.base.barrido);
set(handles.z,'String',handles.base.z);
set(handles.distancia,'String',handles.base.distancia);
set(handles.vmin,'String',handles.base.vmin);
set(handles.vprob,'String',handles.base.vprob);
set(handles.vmax,'String',handles.base.vmax);
```

```
set(handles.frente,'String',handles.base.frente);
set(handles.ti,'String',handles.base.ti);
```

```
set(handles.derrota,'String',ini);
set(handles.rv,'String',ini);
set(handles.horainter,'String',ini);
set(handles.exploradores,'String',ini);
set(handles.valor,'String',ini);
set(handles.arcoab,'String',ini);
set(handles.arcoed,'String',ini);
set(handles.total,'String',ini);
set(handles.avanceinicio,'String',ini);
set(handles.nuevaproab,'String',ini);
set(handles.prob,'String',ini);
set(handles.sc,'String',ini);
set(handles.cc,'String',ini);
```

```
function reporte_Callback(hObject, eventdata, handles)
```

```
% hObject handle to reporte (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of reporte as text
```

```
% str2double(get(hObject,'String')) returns contents of reporte
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
```

```
reporte = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
```

```
if isnan(reporte)
```

```
    set(hObject, 'String', 0);
```

```
    errordlg('Solo debes introducir numeros','Error');
```

```
end
```

```
handles.base.reporte=reporte; %Almacenar en puntero handles pero
del c
```

```
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function reporte_CreateFcn(hObject, eventdata, handles)
```

```
% hObject handle to reporte (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
```

```
    set(hObject,'BackgroundColor','white');
```

```
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function zarpar_Callback(hObject, eventdata, handles)
```

```
% hObject handle to zarpar (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of zarpar as text
```

```
% str2double(get(hObject,'String')) returns contents of zarpar
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
```

```

zarpar = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(zarpar)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.zarpar=zarpar; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function zarpar_CreateFcn(hObject, eventdata, handles)
% hObject handle to zarpar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vmin_Callback(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vmin as text
% str2double(get(hObject,'String')) returns contents of vmin as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmin = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmin)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vmin=vmin; %Almacenar en puntero handles pero del
c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vmin_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vmax_Callback(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vmax as text
% str2double(get(hObject,'String')) returns contents of vmax as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmax = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmax)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vmax=vmax; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vmax_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vprob_Callback(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vprob as text
% str2double(get(hObject,'String')) returns contents of vprob
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vprob = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vprob)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vprob=vprob; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vprob_CreateFcn(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function frente_Callback(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of frente as text
% str2double(get(hObject,'String')) returns contents of frente
as a double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
frente = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(frente)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.frente=frente; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function frente_CreateFcn(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function ti_Callback(hObject, eventdata, handles)
% hObject handle to ti (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ti as text
% str2double(get(hObject,'String')) returns contents of ti as a
double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ti = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ti)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.ti=ti; %Almacenar en puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function ti_CreateFcn(hObject, eventdata, handles)
% hObject handle to ti (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFens
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

% --- Executes on button press in similar.
function similar_Callback(hObject, eventdata, handles)
% hObject handle to similar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
winopen('intercepcion_delante.exe');

```

Pantalla de búsqueda por detrás:

```

function varargout = busqueda_detras(varargin)
% BUSQUEDA_DETRAS M-file for busqueda_detras.fig
% BUSQUEDA_DETRAS, by itself, creates a new
BUSQUEDA_DETRAS or raises the existing
% singleton*.
%
% H = BUSQUEDA_DETRAS returns the handle to a new
BUSQUEDA_DETRAS or the handle to
% the existing singleton*.
%
% BUSQUEDA_DETRAS('CALLBACK',hObject,eventData,handles
,...) calls the local
% function named CALLBACK in BUSQUEDA_DETRAS.M
with the given input arguments.
%
% BUSQUEDA_DETRAS('Property','Value',...) creates a new
BUSQUEDA_DETRAS or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before busqueda_detras_OpeningFunction
gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to busqueda_detras_OpeningFcn
via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

```

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help
busqueda_detras

% Last Modified by GUIDE v2.5 13-Oct-2007 21:31:56

% Begin initialization code - DO NOT EDIT

```
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @busqueda_detras_OpeningFcn, ...
    'gui_OutputFcn', @busqueda_detras_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before busqueda_detras is made visible.
function busqueda_detras_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to busqueda_detras (see
VARARGIN)
```

```
% Choose default command line output for busqueda_detras
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);
```

```
initialize_gui(hObject, handles, false);
```

```
% UIWAIT makes busqueda_detras wait for user response (see
UIRESUME)
% uiwait(handles.figure1);
```

```
% --- Outputs from this function are returned to the command line.
function varargout = busqueda_detras_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
%Colocar Imagen de piedra
fondo = imread('C:\MATLAB7\work\pedras2.jpg'); %Leer imagen
axes(handles.fondo); %Carga la imagen en fondo
axis off;
imshow(fondo); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

```
%Colocar Imagen de fondo
%Carga la imagen de fondo (opcional)
%[x,map]=imread('pedras2.jpg','jpg');
%image(x),colormap(map),axis off,hold on
```

```
%titulos para los cuadros de texto
text(205,20,'Anota los siguientes
datos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(85,59,'Buque
Propio','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweig
ht','Bold','color',[1 1 1]);
text(415,59,'Buque a localizar
','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweight','bol
d','color',[1 1 1]);
text(25,83,'Velocidad','Fontname','Arial','FontSize',9,'Fontangle','Ita
lic','Fontweight','Normal','color',[1 1 1]);
text(93,106,'Nudos','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(372,83,'Reporte','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(370,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(459,83,'Zarpar','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(458,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(557,93,'MV','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(605,116,'grados','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(146,83,'Probabilidad
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(226,106,[' 0 - 100
'],'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','No
rmal','color',[1 1 1]);
text(25,137,'Ancho de barrido
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(95,162,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(370,147,'Distancia','Fontname','Arial','FontSize',9,'Fontangle','I
talic','Fontweight','Normal','color',[1 1 1]);
text(437,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(489,147,'Error de
posición','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(555,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(430,200,'Velocidades
(Nudos)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(387,214,'Vmin','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(473,214,'Vprob','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(562,214,'Vmax','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(149,137,'Según tablas de la normal
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(149,179,'con una probabilidad del:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(216,194,'%','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
```

```

text(25,221,'Si desea usar el
[C]', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(25,234,'obtenido,anotelo:', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(25,246,'en caso contrario
', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(25,258,'deme su valor de C:
', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(372,263,'Longitud Frente de
Exploración', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(507,286,'Millas', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(416,285,'F
=','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(372,309,'Tiempo de intercepción
(hrs)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(230,410,'Resultados
Teóricos', 'Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight','Bold','color',[1 1 1]);
text(55,440,'RV (
grados)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(263,440,'Hora de int. (
Hrs.)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(488,440,'Nº de
exploradores', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(55,500,'Derrota ( S ) Corregida
(Sc)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(139,520,'Millas
Millas', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(488,500,'Nva. Prob.
Contacto', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(481,520,' ( 0 - 100
)%', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(45,570,'Tiempo Total
(Hrs)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(263,570,'Avance Max
(Millas)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(367,599,'Sector
AB', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(488,570,'Avance Min
(Millas)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(592,599,'Sector
CD', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(153,630,'Avance al
principio', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(153,644,'de intercepción ( A ) Factor
Corregido
(Cc)', 'Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);
text(257,672,'Millas
Millas', 'Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','Normal','color',[1 1 1]);

```

```

function velocidadpropio_Callback(hObject, eventdata, handles)
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of velocidadpropio
as text
% str2double(get(hObject,'String')) returns contents of
velocidadpropio as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidadpropio = str2double(NewStrVal); %Transformar a
formato double o de string a numerico
if isnan(velocidadpropio)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velocidadpropio=velocidadpropio; %Almacenar en
puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function velocidadpropio_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function distancia_Callback(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of distancia as text
% str2double(get(hObject,'String')) returns contents of
distancia as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
distancia = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(distancia)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.distancia=distancia; %Almacenar en puntero handles
pero del mvf
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function distancia_CreateFcn(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB

```

```
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function mvi_Callback(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of mvi as text
% str2double(get(hObject,'String')) returns contents of mvi as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
mvi = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(mvi)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.mvi=mvi; %Almacenar en puntero handles pero del
mvi
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function mvi_CreateFcn(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.probabilidad=probabilidad; %Almacenar en puntero
handles pero del probabilidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function ep_Callback(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of ep as text
% str2double(get(hObject,'String')) returns contents of ep as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ep = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ep)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.ep=ep; %Almacenar en puntero handles pero del
velocidad
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function ep_CreateFcn(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function barrido_Callback(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of barrido as text
% str2double(get(hObject,'String')) returns contents of barrido
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
barrido = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(barrido)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.barrido=barrido; %Almacenar en puntero handles
pero del barrido
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function barrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of z as text
% str2double(get(hObject,'String')) returns contents of z as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.z=z; %Almacenar en puntero handles pero del c
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject handle to tablas (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
open('C:\MATLAB7\work\tablas.pdf');
```

```
% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
%----- VARIABLES PARA DETERMINAR DERROTAS (S) La
cual no aparecerá en pantalla-----
Z=handles.base.z;
W=handles.base.barrido;
S=1.253*(W/Z);
set(handles.derrota,'string',S);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----
```

```
%-----AVANCE DEL ENEMIGO SEGUN HORA DE REPORTE
Y HORA DE ZARPAR-----
if handles.base.reporte<handles.base.zarpar
    tm= hms2hr(handles.base.zarpar)- hms2hr(handles.base.reporte);
    else
        tm= hms2hr(handles.base.reporte)-
hms2hr(handles.bese.zarpar);
end
```

```
avanceinicio = (handles.base.vmin * tm);
set(handles.avanceinicio,'string',avanceinicio);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----
```

```
%-----TIMEPO TOTAL-----
-----
Tt= tm + handles.base.ti;
set(handles.total,'string',Tt);
guidata(hObject, handles);
%-----FIN-----
```

```
%-----AVANCE MAXIMO-----
-----
arcoab = Tt * handles.base.vmax;
set(handles.arcoab,'string',arcoab);
guidata(hObject, handles);
%-----FIN-----
```

```
%-----AVANCE MINIMO-----
-----
arcood= Tt * handles.base.vmin;
```

```

set(handles.arccod,'string',arccod);
guidata(hObject, handles);
%-----FIN-----

%-----NUMERO DE EXPLORADORES -----
F = handles.base.frente;
if S == W
    N = ceil(F / S);
    Sc = F/N;
    Cc = W / Sc;
    set (handles.exploradores,'string',N);
    set (handles.sc,'string',Sc);
    set (handles.cc,'string',Cc);
    guidata(hObject, handles);
else
    N = ceil(((F - W) / S) + 1);
    Sc = (F - W) / (N - 1);
    Cc = W / Sc;
    set (handles.exploradores,'string',N);
    set (handles.sc,'string',Sc);
    set (handles.cc,'string',Cc);
    guidata(hObject, handles);
end

guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----NUEVA PROBABIIDAD-----
nueva=normcdf(Cc,0,1);
probanueva= ((nueva - 0.5)*2)*100;
set(handles.nuevaprob,'String',probanueva);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

%-----HORA INTERSECCIÓN-----
ti = handles.base.ti;
hora= hms2hr(handles.base.reporte)+ ti;
horainter = time2str(hora);
set(handles.horainter,'String',horainter);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

function derrota_Callback(hObject, eventdata, handles)
% hObject handle to derrota (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of derrota as text
% str2double(get(hObject,'String')) returns contents of derrota
as a double

% --- Executes during object creation, after setting all properties.
function derrota_CreateFcn(hObject, eventdata, handles)
% hObject handle to derrota (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));

```

end

```

function exploradores_Callback(hObject, eventdata, handles)
% hObject handle to exploradores (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of exploradores as
text
% str2double(get(hObject,'String')) returns contents of
exploradores as a double

```

```

% --- Executes during object creation, after setting all properties.
function exploradores_CreateFcn(hObject, eventdata, handles)
% hObject handle to exploradores (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

Pd=handles.base.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

```

```

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in limpiar.
function limpiar_Callback(hObject, eventdata, handles)
% hObject handle to limpiar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

initialize_gui(gcbof, handles, true);

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are we are just re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles,'base') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.base.radio = 0;
handles.base.mvi = 0;
handles.base.reporte = 0;
handles.base.probabilidad = 0;
handles.base.barrido = 0;
handles.base.velocidad = 0;
handles.base.z = 0;
handles.base.reporte = ('0000');
handles.base.zarpar = ('0000');
handles.base.distancia = 0;
handles.base.vmin = 0;
handles.base.vprob = 0;
handles.base.vmax = 0;
handles.base.frente = 0;
handles.base.ti = 0;

set(handles.velocidadpropio,'String',handles.base.radio);
set(handles.reporte,'String',handles.base.reporte);
set(handles.zarpar,'String',handles.base.zarpar);
set(handles.mvi,'String',handles.base.mvi);
set(handles.probabilidad,'String',handles.base.probabilidad);
set(handles.ep,'String',handles.base.velocidad);
set(handles.barrido,'String',handles.base.barrido);
set(handles.z,'String',handles.base.z);
set(handles.distancia,'String',handles.base.distancia);
set(handles.vmin,'String',handles.base.vmin);
set(handles.vprob,'String',handles.base.vprob);
set(handles.vmax,'String',handles.base.vmax);
set(handles.frente,'String',handles.base.frente);
set(handles.ti,'String',handles.base.ti);

set(handles.derrota,'String',ini);
set(handles.rv,'String',ini);
set(handles.horainter,'String',ini);
set(handles.exploradores,'String',ini);
set(handles.valor,'String',ini);
set(handles.arcoab,'String',ini);

```

```

set(handles.arcoab,'String',ini);
set(handles.total,'String',ini);
set(handles.avanceinicio,'String',ini);
set(handles.nuevapro,'String',ini);
set(handles.prob,'String',ini);
set(handles.sc,'String',ini);
set(handles.cc,'String',ini);

```

```

function nuevapro_Callback(hObject, eventdata, handles)
% hObject handle to nuevapro (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of nuevapro as text
% str2double(get(hObject,'String')) returns contents of
nuevapro as a double

```

```

% --- Executes during object creation, after setting all properties.
function nuevapro_CreateFcn(hObject, eventdata, handles)
% hObject handle to nuevapro (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function reporte_Callback(hObject, eventdata, handles)
% hObject handle to reporte (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of reporte as text
% str2double(get(hObject,'String')) returns contents of reporte
as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
reporte = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(reporte)
    set(hObject, 'String', '0');
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.reporte=reporte; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function reporte_CreateFcn(hObject, eventdata, handles)
% hObject handle to reporte (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.

```

```

if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function zarpar_Callback(hObject, eventdata, handles)
% hObject handle to zarpar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of zarpar as text
% str2double(get(hObject,'String')) returns contents of zarpar
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
zarpar = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(zarpar)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.zarpar=zarpar; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function zarpar_CreateFcn(hObject, eventdata, handles)
% hObject handle to zarpar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vmin_Callback(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vmin as text
% str2double(get(hObject,'String')) returns contents of vmin as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmin = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmin)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vmin=vmin; %Almacenar en puntero handles pero del
c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vmin_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vmax_Callback(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vmax as text
% str2double(get(hObject,'String')) returns contents of vmax as
a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmax = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmax)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vmax=vmax; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vmax_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function vprob_Callback(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of vprob as text
% str2double(get(hObject,'String')) returns contents of vprob
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vprob = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vprob)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vprob=vprob; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function vprob_CreateFcn(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function frente_Callback(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of frente as text
% str2double(get(hObject,'String')) returns contents of frente
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
frente = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(frente)
    set(hObject, 'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.frente=frente; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function frente_CreateFcn(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function rv_Callback(hObject, eventdata, handles)
% hObject handle to rv (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of rv as text
% str2double(get(hObject,'String')) returns contents of rv as a
double

```

```

% --- Executes during object creation, after setting all properties.
function rv_CreateFcn(hObject, eventdata, handles)
% hObject handle to rv (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function horainter_Callback(hObject, eventdata, handles)
% hObject handle to horainter (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of horainter as text
% str2double(get(hObject,'String')) returns contents of
horainter as a double

```

```

% --- Executes during object creation, after setting all properties.
function horainter_CreateFcn(hObject, eventdata, handles)
% hObject handle to horainter (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function ti_Callback(hObject, eventdata, handles)
% hObject handle to ti (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of ti as text
% str2double(get(hObject,'String')) returns contents of ti as a
double

```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ti = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ti)
    set(hObject, 'String', 0);
    error('Solo debes introducir numeros','Error');
end
handles.base.ti=ti; %Almacenar en puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function ti_CreateFcn(hObject, eventdata, handles)
% hObject handle to ti (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function total_Callback(hObject, eventdata, handles)
% hObject handle to total (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of total as text
% str2double(get(hObject,'String')) returns contents of total as
a double
```

```
% --- Executes during object creation, after setting all properties.
function total_CreateFcn(hObject, eventdata, handles)
% hObject handle to total (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function arcoab_Callback(hObject, eventdata, handles)
% hObject handle to arcoab (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of arcoab as text
```

```
% str2double(get(hObject,'String')) returns contents of arcoab
as a double
```

```
% --- Executes during object creation, after setting all properties.
function arcoab_CreateFcn(hObject, eventdata, handles)
% hObject handle to arcoab (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function arccod_Callback(hObject, eventdata, handles)
% hObject handle to arccod (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of arccod as text
% str2double(get(hObject,'String')) returns contents of arccod
as a double
```

```
% --- Executes during object creation, after setting all properties.
function arccod_CreateFcn(hObject, eventdata, handles)
% hObject handle to arccod (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function avanceinicio_Callback(hObject, eventdata, handles)
% hObject handle to avanceinicio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of avanceinicio as
text
% str2double(get(hObject,'String')) returns contents of
avanceinicio as a double
```

```
% --- Executes during object creation, after setting all properties.
function avanceinicio_CreateFcn(hObject, eventdata, handles)
% hObject handle to avanceinicio (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in similar.
function similar_Callback(hObject, eventdata, handles)
% hObject handle to similar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

winopen('intercepcion_detras.exe');
```

Pantalla de búsqueda por un flanco:

```
function varargout = busqueda_flanco(varargin)
% BUSQUEDA_FLANCO M-file for busqueda_flanco.fig
% BUSQUEDA_FLANCO, by itself, creates a new
BUSQUEDA_FLANCO or raises the existing
% singleton*.
%
% H = BUSQUEDA_FLANCO returns the handle to a new
BUSQUEDA_FLANCO or the handle to
% the existing singleton*.
%
% BUSQUEDA_FLANCO('CALLBACK',hObject,eventData,handles
,...) calls the local
% function named CALLBACK in BUSQUEDA_FLANCO.M
with the given input arguments.
%
% BUSQUEDA_FLANCO('Property','Value',...) creates a new
BUSQUEDA_FLANCO or raises the
% existing singleton*. Starting from the left, property value
pairs are
% applied to the GUI before busqueda_flanco_OpeningFunction
gets called. An
% unrecognized property name or invalid value makes property
application
% stop. All inputs are passed to busqueda_flanco_OpeningFcn
via varargin.
%
% *See GUI Options on GUIDE's Tools menu. Choose "GUI
allows only one
% instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.
```

```
% Edit the above text to modify the response to help
busqueda_flanco
```

```
% Last Modified by GUIDE v2.5 13-Oct-2007 21:33:43
```

```
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @busqueda_flanco_OpeningFcn, ...
    'gui_OutputFcn', @busqueda_flanco_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before busqueda_flanco is made visible.
function busqueda_flanco_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to busqueda_flanco (see
VARARGIN)
```

```
% Choose default command line output for busqueda_flanco
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);
```

```
initialize_gui(hObject, handles, false);
```

```
% UIWAIT makes busqueda_flanco wait for user response (see
UIRESUME)
% uiwait(handles.figure1);
```

```
% --- Outputs from this function are returned to the command line.
function varargout = busqueda_flanco_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
%Colocar Imagen de piedra
fondo = imread('C:\MATLAB7\work\piedras2.jpg'); %Leer imagen
axes(handles.fondo); %Carga la imagen en fondo
axis off;
imshow(fondo); %Presenta la imagen
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

```

%Colocar Imagen de fondo
%Carga la imagen de fondo (opcional)
%[x_map]=imread('pedras2.jpg','jpg');
%image(x,colormap(map),axis off,hold on

%titulos para los cuadros de texto
text(205,20,'Anota los siguientes
datos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontweight
','Bold','color',[1 1 1]);
text(85,59,'Buque
Propio','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweig
ht','Bold','color',[1 1 1]);
text(415,59,'Buque a localizar
','Fontname','Arial','FontSize',11,'Fontangle','Italic','Fontweight','bol
d','color',[1 1 1]);
text(25,83,'Velocidad','Fontname','Arial','FontSize',9,'Fontangle','Ita
lic','Fontweight','Normal','color',[1 1 1]);
text(93,106,'Nudos','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(372,83,'Reporte','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(370,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(459,83,'Zarpar','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(458,97,'(Hrs,min)','Fontname','Arial','FontSize',8,'Fontangle','It
alic','Fontweight','Normal','color',[1 1 1]);
text(557,93,'MV','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(605,116,'grados','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(146,83,'Probabilidad
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(226,106,[' 0 - 100 ]
%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','No
rmal','color',[1 1 1]);
text(25,137,'Ancho de barrido
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(95,162,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(370,147,'Distancia','Fontname','Arial','FontSize',9,'Fontangle','I
talic','Fontweight','Normal','color',[1 1 1]);
text(437,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(489,147,'Error de
posición','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(555,172,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(430,200,'Velocidades
(Nudos)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(387,214,'Vmin','Fontname','Arial','FontSize',9,'Fontangle','Italic
','Fontweight','Normal','color',[1 1 1]);
text(473,214,'Vprob','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(562,214,'Vmax','Fontname','Arial','FontSize',9,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(149,137,'Según tablas de la normal
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(149,179,'con una probabilidad del:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(216,194,'%','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(25,221,'Si desea usar el
[C]','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);

text(25,234,'obtenido,anotelo:','Fontname','Arial','FontSize',9,'Fonta
ngle','Italic','Fontweight','Normal','color',[1 1 1]);
text(25,246,'en caso contrario
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(25,258,'deme su valor de C:
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
%text(372,263,',','Fontname','Arial','FontSize',9,'Fontangle','Italic','F
ontweight','Normal','color',[1 1 1]);
text(566,286,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(372,285,'Tiempo de
búsqueda','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(372,309,'Tiempo de intercepción
(hrs)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(230,410,'Resultados
Teóricos','Fontname','Arial','FontSize',13,'Fontangle','Italic','Fontwe
ight','Bold','color',[1 1 1]);
text(55,440,'RV (
grados)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(263,440,'Tiempo muerto (
Hrs.)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(488,440,'Nº de
exploradores','Fontname','Arial','FontSize',9,'Fontangle','Italic','Font
weight','Normal','color',[1 1 1]);
text(25,500,'Derrota (S) Corregida
(Sc)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(172,527,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(488,500,'Nva. Prob.
Contacto','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontwei
ght','Normal','color',[1 1 1]);
text(592,527,'( 0 - 100
)%','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','N
ormal','color',[1 1 1]);
text(263,500,'Pto. de intercepción ( A
)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(367,527,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(10,570,'Tiempo Total (Hrs) Nuevo Total
(Hrs)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(263,570,'Avance Max
(Millas)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(367,599,'Sector
AB','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(488,570,'Avance Min
(Millas)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweig
ht','Normal','color',[1 1 1]);
text(592,599,'Sector
CD','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(75,630,'Distancia al punto
','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);
text(25,645,'( A ) "Q"
"R"',Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','
Normal','color',[1 1 1]);
text(273,644,'Frente de exploracion (F) Nuevo Frente de
Exploración (F'
)','Fontname','Arial','FontSize',9,'Fontangle','Italic','Fontweight','Nor
mal','color',[1 1 1]);

```

```
text(217,672,'Millas','Fontname','Arial','FontSize',8,'Fontangle','Itali
c','Fontweight','Normal','color',[1 1 1]);
text(373,672,'Millas
Millas','Fontname','Arial','FontSize',8,'Fontangle','Italic','Fontweight
','Normal','color',[1 1 1]);
```

```
function velocidadpropio_Callback(hObject, eventdata, handles)
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of velocidadpropio
as text
% str2double(get(hObject,'String')) returns contents of
velocidadpropio as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
velocidadpropio = str2double(NewStrVal); %Transformar a
formato double o de string a numerico
if isnan(velocidadpropio)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.velocidadpropio=velocidadpropio; %Almacenar en
puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function velocidadpropio_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidadpropio (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function distancia_Callback(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of distancia as text
% str2double(get(hObject,'String')) returns contents of
distancia as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
distancia = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(distancia)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.distancia=distancia; %Almacenar en puntero handles
pero del mvf
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function distancia_CreateFcn(hObject, eventdata, handles)
% hObject handle to distancia (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function mvi_Callback(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of mvi as text
% str2double(get(hObject,'String')) returns contents of mvi as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
mvi = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(mvi)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.mvi=mvi; %Almacenar en puntero handles pero del
mvi
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function mvi_CreateFcn(hObject, eventdata, handles)
% hObject handle to mvi (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function probabilidad_Callback(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of probabilidad as
text
% str2double(get(hObject,'String')) returns contents of
probabilidad as a double
```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
probabilidad = str2double(NewStrVal); %Transformar a formato
double o de string a numerico
if isnan(probabilidad)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.probabilidad=probabilidad; %Almacenar en puntero
handles pero del probabilidad
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function probabilidad_CreateFcn(hObject, eventdata, handles)
% hObject handle to probabilidad (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function ep_Callback(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of ep as text
% str2double(get(hObject,'String')) returns contents of ep as a
double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ep = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ep)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.ep=ep; %Almacenar en puntero handles pero del
velocidad
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function ep_CreateFcn(hObject, eventdata, handles)
% hObject handle to ep (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function barrido_Callback(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of barrido as text
% str2double(get(hObject,'String')) returns contents of barrido
as a double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
barrido = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(barrido)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.barrido=barrido; %Almacenar en puntero handles
pero del barrido
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function barrido_CreateFcn(hObject, eventdata, handles)
% hObject handle to barrido (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

```

```

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

function z_Callback(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of z as text
% str2double(get(hObject,'String')) returns contents of z as a
double

```

```

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
z = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(z)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.z=z; %Almacenar en puntero handles pero del c
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

% --- Executes during object creation, after setting all properties.
function z_CreateFcn(hObject, eventdata, handles)
% hObject handle to z (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB

```

```

% handles empty - handles not created until after all CreateFncs
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

% --- Executes on button press in tablas.
function tablas_Callback(hObject, eventdata, handles)
% hObject handle to tablas (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

open ('C:\MATLAB7\work\tablas.pdf');

% --- Executes on button press in buscar.
function buscar_Callback(hObject, eventdata, handles)
% hObject handle to buscar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

%----- VARIABLES PARA DETERMINAR DERROTAS (S) La
cual no aparecerá en pantalla-----
Z=handles.base.z;
W=handles.base.barrido;
S=1.253*(W/Z);
set(handles.derrota,'string',S);
guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN -----

%-----AVANCE DEL ENEMIGO SEGUN HORA DE REPORTE
Y HORA DE ZARPAR-----
if handles.base.reporte<handles.base.zarpar
    tm= hms2hr(handles.base.zarpar)- hms2hr(handles.base.reporte);
else
    tm=
        hms2hr(handles.base.reporte)-
hms2hr(handles.bese.zarpar);
end

    avanceinicio = (handles.base.vprob * tm);
    set(handles.avanceinicio,'string',avanceinicio);
    guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----TIMEPO TOTAL-----
Tt= tm + handles.base.ti;
set(handles.total,'string',Tt);
guidata(hObject, handles);
%-----FIN-----

%-----AVANCE MAXIMO ARCO AB-----
arcoab = Tt * handles.base.vmax;
set(handles.arcoab,'string',arcoab);
guidata(hObject, handles);
%-----FIN-----

%-----AVANCE MINIMO ARCO CD-----
arcocd= Tt * handles.base.vmin;

```

```

set(handles.arcocd,'string',arcocd);
guidata(hObject, handles);
%-----FIN-----

%-----FRENTE DE EXPLORACIÓN-----
ep = handles.base.ep;
F = ( arcoab + ep ) - ( arcocd - ep );
set(handles.f,'string',F);
guidata(hObject, handles);
%-----FIN-----

%-----PUNTO DE INTERCEPCIÓN ( A )-----
puntoa = handles.base.vprob * Tt;
set(handles.puntoa,'string',puntoa);
guidata(hObject, handles);
%-----FIN-----

%-----TOTAL NUEVO-----
nuevototal= tm + handles.base.ti + handles.base.frente;
set(handles.nuevototal,'string',nuevototal);
guidata(hObject, handles);
%-----FIN-----

%-----DISTANCIA Q-----
q=handles.base.vmin * nuevototal;
set(handles.q,'string',q);
guidata(hObject, handles);
%-----FIN-----

%-----DISTANCIA R-----
r = handles.base.vmax * nuevototal;
set(handles.r,'string',r);
guidata(hObject, handles);
%-----FIN-----

%-----NUEVO FRENTE-----
NF = (r + ep) - (q - ep);
set(handles.nuevofrente,'string',NF);
guidata(hObject, handles);
%-----FIN-----

%-----NUMERO DE EXPLORADORES -----
%F = handles.base.frente;
if S == W
    N = ceil(NF / S);
    Sc = NF/N;
    Cc = W / Sc;
    set(handles.exploradores,'string',N);
    set(handles.sc,'string',Sc);
    guidata(hObject, handles);
else
    N = ceil(((NF - W) / S) + 1);
    Sc = (NF - W) / (N - 1);
    Cc = W / Sc;
    set(handles.exploradores,'string',N);
    set(handles.sc,'string',Sc);
    guidata(hObject, handles);
end

guidata(hObject,handles); %Salvar datos de la aplicación
%-----FIN-----

%-----NUEVA PROBABIIDAD-----
nueva=normcdf(Cc,0,1);

```

```

probanueva= ((nueva - 0.5)*2)*100;
set(handles.nuevapro, 'String',probanueva);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

%-----HORA INTERSECCIÓN-----
%hora= hms2hr(handles.base.reporte)+ ti;
%horainter = time2str(hora);
set(handles.horainter,'String',tm);
guidata(hObject,handles); %Salvar datos de la aplicaciónguidata
%-----FIN-----

```

```

% --- Executes on button press in cubo.
function cubo_Callback(hObject, eventdata, handles)
% hObject handle to cubo (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

Pd=handles.base.probabilidad;
P = Pd/100;
total = P/2 + 0.5;
C=norminv(total, 0, 1);
set(handles.valor,'String',C);
set(handles.prob,'String',Pd);
guidata(hObject,handles); %Salvar datos de la aplicación

```

```

function valor_Callback(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of valor as text
% str2double(get(hObject,'String')) returns contents of valor as
a double

```

```

% --- Executes during object creation, after setting all properties.
function valor_CreateFcn(hObject, eventdata, handles)
% hObject handle to valor (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

```

```

% --- Executes on button press in limpiar.
function limpiar_Callback(hObject, eventdata, handles)
% hObject handle to limpiar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

initialize_gui(gcbf, handles, true);

```

```

%-----
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it
means
% we are re-initializing a GUI by calling it from the
cmd line
% while it is up. So, bail out as we don't want to reset the data.
if isfield(handles,'base') && ~isreset
    return;
end
%Limpiar pantalla
ini=char(' ');
%-----
handles.base.radio = 0;
handles.base.mvi = 0;
handles.base.reporte = 0;
handles.base.probabilidad = 0;
handles.base.barrido = 0;
handles.base.velocidad = 0;
handles.base.z = 0;
handles.base.reporte = ('0000');
handles.base.zarpar = ('0000');
handles.base.distancia = 0;
handles.base.vmin = 0;
handles.base.vprob = 0;
handles.base.vmax = 0;
handles.base.frente = 0;
handles.base.ti = 0;

```

```

set(handles.velocidadpropio,'String',handles.base.radio);
set(handles.reporte,'String',handles.base.reporte);
set(handles.zarpar,'String',handles.base.zarpar);
set(handles.mvi,'String',handles.base.mvi);
set(handles.probabilidad,'String',handles.base.probabilidad);
set(handles.ep,'String',handles.base.velocidad);
set(handles.barrido,'String',handles.base.barrido);
set(handles.z,'String',handles.base.z);
set(handles.distancia,'String',handles.base.distancia);
set(handles.vmin,'String',handles.base.vmin);
set(handles.vprob,'String',handles.base.vprob);
set(handles.vmax,'String',handles.base.vmax);
set(handles.frente,'String',handles.base.frente);
set(handles.ti,'String',handles.base.ti);

```

```

set(handles.derrota,'String',ini);
set(handles.rv,'String',ini);
set(handles.horainter,'String',ini);
set(handles.exploradores,'String',ini);
set(handles.valor,'String',ini);
set(handles.arcoab,'String',ini);
set(handles.arcoed,'String',ini);
set(handles.total,'String',ini);
set(handles.avanceinicio,'String',ini);
set(handles.nuevapro, 'String',ini);
set(handles.prob,'String',ini);
set(handles.f,'String',ini);
set(handles.puntoa,'String',ini);
set(handles.sc,'String',ini);
set(handles.nuevototal,'String',ini);
set(handles.q,'String',ini);
set(handles.r,'String',ini);
set(handles.nuevofrente,'String',ini);

```

```

function reporte_Callback(hObject, eventdata, handles)
% hObject handle to reporte (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB

```

```

% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of reporte as text
% str2double(get(hObject,'String')) returns contents of reporte
as a double
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
reporte = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(reporte)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.reporte=reporte; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function reporte_CreateFcn(hObject, eventdata, handles)
% hObject handle to reporte (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function zarpar_Callback(hObject, eventdata, handles)
% hObject handle to zarpar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of zarpar as text
% str2double(get(hObject,'String')) returns contents of zarpar
as a double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
zarpar = str2double(NewStrVal); %Transformar a formato double
o de string a numerico
if isnan(zarpar)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.zarpar=zarpar; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function zarpar_CreateFcn(hObject, eventdata, handles)
% hObject handle to zarpar (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');

```

```

else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function vmin_Callback(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of vmin as text
% str2double(get(hObject,'String')) returns contents of vmin as
a double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmin = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmin)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vmin=vmin; %Almacenar en puntero handles pero del
c
guidata(hObject,handles); %Salvar datos de la aplicación

% --- Executes during object creation, after setting all properties.
function vmin_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmin (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else

set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end

function vmax_Callback(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of vmax as text
% str2double(get(hObject,'String')) returns contents of vmax as
a double

NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vmax = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vmax)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end

```

```
handles.base.vmax=vmax; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function vmax_CreateFcn(hObject, eventdata, handles)
% hObject handle to vmax (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function vprob_Callback(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of vprob as text
% str2double(get(hObject,'String')) returns contents of vprob
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
vprob = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(vprob)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.vprob=vprob; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function vprob_CreateFcn(hObject, eventdata, handles)
% hObject handle to vprob (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function frente_Callback(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of frente as text
% str2double(get(hObject,'String')) returns contents of frente
as a double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
frente = str2double(NewStrVal); %Transformar a formato double o
de string a numerico
if isnan(frente)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.frente=frente; %Almacenar en puntero handles pero
del c
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
function frente_CreateFcn(hObject, eventdata, handles)
% hObject handle to frente (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

```
function ti_Callback(hObject, eventdata, handles)
% hObject handle to ti (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of ti as text
% str2double(get(hObject,'String')) returns contents of ti as a
double
```

```
NewStrVal=get(hObject,'String'); %Almacenar valor ingresado
ti = str2double(NewStrVal); %Transformar a formato double o de
string a numerico
if isnan(ti)
    set(hObject,'String', 0);
    errordlg('Solo debes introducir numeros','Error');
end
handles.base.ti=ti; %Almacenar en puntero handles pero del radio
guidata(hObject,handles); %Salvar datos de la aplicación
```

```
% --- Executes during object creation, after setting all properties.
```

```
function ti_CreateFcn(hObject, eventdata, handles)
```

```
% hObject handle to ti (see GCBO)
```

```
% eventdata reserved - to be defined in a future version of
MATLAB
```

```
% handles empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on
Windows.
```

```
% See ISPC and COMPUTER.
```

```
if ispc
    set(hObject,'BackgroundColor','white');
else
```

```
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackground
Color'));
end
```

end

```
% --- Executes on button press in similar.  
function similar_Callback(hObject, eventdata, handles)  
% hObject    handle to similar (see GCBO)  
% eventdata  reserved - to be defined in a future version of  
MATLAB  
% handles    structure with handles and user data (see GUIDATA)  
  
winopen('intercepcion_flanco.exe'); %abre la ventana para la  
simulación de flash.
```


Bibliografía

Naval Operations Analysis

Daniel H. Wagner, W Charles Mylander
Naval Institute Press. Third Edition
Unites Estates of America. 1999

Toma de decisiones (un enfoque integrado)

David Jennings, Stuart Wattam
Edit. CESCA, Primera Edición
México, 2000.

Técnicas para la toma de decisiones

Ernesto Mercado Ramírez
Edit. Limusa, Grupo Noriega Editores, Primera Edición
México 1991.

Búsqueda y Detección

Secretaria de Marina, Dirección General de Organización y Sistemas, Dirección de Investigación de Operaciones
junio 1990.

Análisis de Operaciones Navales

Escuela Naval Militar
Marin 1980, Publicación 317.

<http://docencia.50webs.com/simula10.htm>

Fundamentos de Radar

Escuela Naval Militar
Marin, Publicación 459.

Apuntes Escolares

Maestra en Ingenieria Francis Irene. 2007.