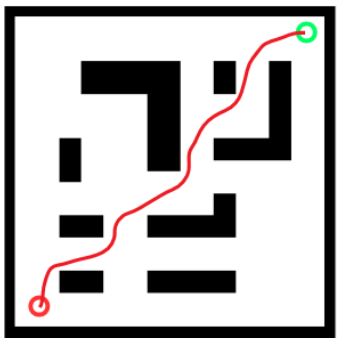
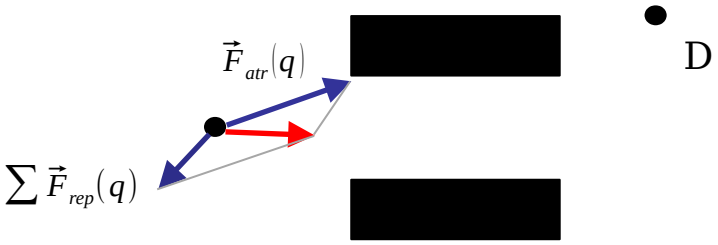
Se pretende desarrollar un módulo de navegación local o reactiva capaz de evitar obstáculos en tiempo real a partir de las lecturas de los sensores de a bordo de un robot móvil, mientras éste navega desde una posición inicial hasta su destino.



Dado que se requiere un módulo de navegación reactiva que funcione en tiempo real, que calcule el movimiento del robot en tiempo de ejecución, se plantea la implementación del método de campos potenciales en su versión más apropiada para tiempo real. Es decir, calcular únicamente las fuerzas de atracción y repulsión sobre el robot y no sobre todo el entorno que le rodea. Por tanto, dada una distribución de obstáculos y lecturas del sensor, en cada iteración tendrán que calcularse únicamente las fuerzas de atracción (al objetivo) y repulsión (de los obstáculos) en la ubicación donde se encuentre el robot en cada instante:

$$\vec{F}_{res}(q) = \vec{F}_{atr}(q) + \sum \vec{F}_{rep}(q)$$



Como primera propuesta se plantea realizar la implementación del algoritmo en Matlab, de forma que la información del entorno local que captan los sensores del robot se carguen a partir de una imagen. Esta imagen y un *script* a modo de plantilla se proporcionan con el material de prácticas. Los archivos que se proporcionan son:

- ***mapa1_150.png***. Imagen que simula un mapa con los obstáculos del entorno.
- ***plantilla_campos_potenciales.m***. Como se indica en los comentarios de dicha plantilla hay que completar el movimiento del robot calculado a partir de las fuerzas a las que está sometido por el destino y los obstáculos y, antes de realizar estos cálculos en el bucle, crear una lista con los obstáculos.

Se pide:

1. Completar el script denominado *plantilla* para conseguir la navegación reactiva del robot según el método de campos potenciales.

2. Realizar varios experimentos con diferentes orígenes y destinos. Cambiar los parámetros del método, como α y β, o cualquier otro, para tratar de mejorar la situación. Poner el destino y origen en una situación de mínimo local (situación de trampa local), ¿es posible que el método pueda encontrar la solución sin modificarlo y cambiando solo los parámetros anteriormente mencionados?.