Trabajo de Computación Distribuida para Sistemas Multirrobot: Alineación de Grupos de Robots Usando Métodos de Potencial

Iñaki Rañó

1. Introducción

Este trabajo consiste en la implementación y evaluación de un comportamiento de movimiento colectivo (*flocking*) para sistemas multirrobot usando métodos de potencial. Los robots del enjambre se inicializarán aleatoriamente (posición y velocidad) en el entorno y se moverán en línea recta con velocidad máxima cambiando su dirección de manera aleatoria hasta que detecten algún robot en su vecindario. Hay dos tipos de robots (de colores rojo y azul) que deberán agruparse según su color, mientras que robots de colores diferentes se intentarán alejar entre ellos. Los robots que formen un grupo de dos o más deberán moverse en la misma dirección a velocidad máxima, y con cierta probabilidad de que cada uno cambie un poco, i.e. individualmente, su dirección de movimiento (aleatoriamente) mientras intenta mantener una cierta distancia con sus vecinos. La topología del entorno hace que la implementación del comportamiento requiera cierto cuidado, especialmente cerca de los "límites" del entorno. Por otro lado será necesario implementar una función de evaluación del estado del enjambre que mida alguna característica apropiada de los enjambres para poder evaluar numéricamente el comportamiento. Dicha función puede evaluar la dispersión de cada uno de los enjambres y/o el número de grupos que se forma con su respectivo tamaño, entre otras características.

El código resultante, así como un informe de máximo 6 páginas deberá ser entregado en el plazo indicado a través del campus virtual.

1.1. Topología del entorno

Como en la práctica de agrupamiento, el entorno tiene una topología toroidal en la que cada lado del cuadrado corresponde con el lado opuesto como se puede ver en la figura 1.1. Las flechas en las aristas del cuadrado (arriba y abajo, derecha e izquierda) representan la correspondencia entre puntos. Si se junta primero la arista superior con la inferior el entorno se convierte en un cilindro como se puede ver en la imágen central de la figura. La distancia entre los puntos A y B ya no es calculada a través de una línea recta sino que se calcula a lo largo de la superficie del cilindro. Si ahora se juntan los círculos que definen los límites del cilindro (lados izquierdo y derecho del cuadrado original) se llega a una superficie toroidal, que es donde se moverán los

robots. Como se puede ver en la imágen derecha la distancia entre los puntos A y B en la superficie toroidal es mucho menor que la calculada inicialmente.

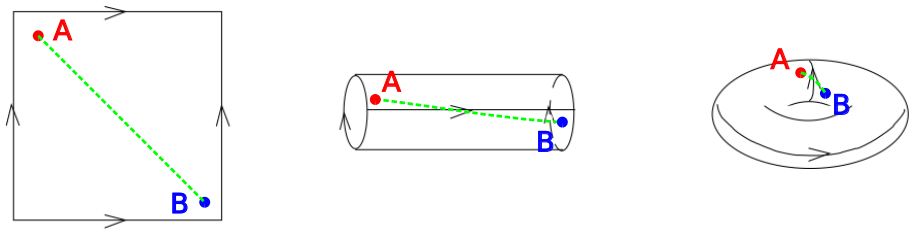

Figura 1: Topología del entorno simulado.

Esta topología del entorno afecta a la implementación del comportamiento de los robots del ejambre como veremos más adelante. Además de la distancia entre dos puntos del entorno, los vectores definidos como diferencias de dos puntos también se ven afectados. Por ejemplo, si se define el vector ($\vec{AB}$) que va del punto A al punto B como $\vec{AB} = B - A$, en el caso del entorno toroidal de la figura el vector, al representarlo en el plano (imágen izquierda), saldría del cuadrado por la esquina superior izquierda y llegaría a B por la esquina inferior derecha. Por lo tanto la definición del vector $\vec{AB} = B - A$ no es adecuada para este entorno. Sin embargo, si conocemos la distancia (d_{AB}) entre los puntos A y B y el ángulo que forma el segmento $A - B$ con el eje x del sistema de referencia del entorno α_{AB} el vector se puede obtener como $\vec{AB} = d_{AB}[\cos \alpha_{AB}, \sin \alpha_{AB}]$.

Otro problema que aparece en este entorno es el cálculo de la posición media entre dos o más puntos. Por ejemplo si se intenta calcular la posición média entre A y B en la representación plana del entorno de la figura (imágen izquierda) el punto médio estará cerca del origen (en el centro del cuadrado). Sin embargo, dada la topología del entorno, el punto medio entre A y B está en realidad cerca de la esquina superior izquierda. Esto también ocurre al calcular promedios de variables angulares, como por ejemplo direcciones de vectores. El apéndice A explica cómo se pueden calcular promedios de variables periodicas a través de una definición matemática de lo que es en realidad un promedio[1]. La librería para la simulación del enjambre proporciona funcionalidad para manejar este entorno, pero es convemiente entender por qué esta funcionalidad es necesaria y, hasta cierto punto, qué es lo que hace.

2. Comportamiento de *Flocking* del Enjambre

El comportamiento de *focking* se puede implementar de diversas maneras, aunque en este trabajo nos centraremos en métodos de potencial. Una opción típica para generar comportamiento de *flocking* es usar Boids, aunque, como veremos, se puede implementar con otros métodos. Puesto que los robots deben alinear sus velocidades, mantenerse a cierta distancia de los vecinos del mismo color y alejarse de los vecinos de distinto color habrá que calcular diferentes contribuciones a la velocidad y combinarlas con pesos a asignar de forma empírica. Para poder seleccionar los pesos de la

[1]TL;DR El punto medio μ de un conjunto de puntos $\{x_1, x_2, \cdots, x_n\}$ es el punto en el cual la suma de las distancias a los puntos del conjunto ($\sum_{i=1}^{n} d(x_i, \mu)$) es mínima.

combinación hay que implementar una función que evalue el comportamiento de los enjambres, de forma que se tenga un valor (o valores) cuantitativos del comportamiento.

2.1. Comportamiento de Alineación

Para implementar el comportamiento de alineación se puede hacer que la velocidad de un robot sea la velocidad promedio de los robots del mismo color en su entorno cercano, i.e. los robots que puede percibir. En el cálculo de esta velocidad promedio puede hacerse de varias formas (media vectorial o media de águlos y longitudes), y puede incluirse la velocidad propia o no. Dado que las velocidades iniciales se distribuyen aleatoriamente una media directa entre vectores dará una velocidad promedio cercana a cero, lo que violaría el requerimiento de que los grupos se muevan con velocidad máxima, por lo que es necesario usar algún mecanismo para asegurarse que la velocidad del enjambre sea la velocidad máxima.

2.2. Mantener la Distancia a los Vecinos

Para mantener la distancia a sus vecinos los Boids implementan los comportamientos de separación y cohexión, que combinados hacen que cada robot se mantenga a cierta distancia de sus vecinos. Alternativamente puede añadirse a la velocidad de cada robot un término similar a un controlador proporcional que haga que el robot se acerque a un vecino al estilo *Shucker et al*[2]. Este término ha de promediarse para todos los vecinos dentro del entorno de percepción del robot, y dado que el trabajo citado usa un modelo integrador doble (vs. el modelo integrador del simulador) es necesario adaptar el cálculo de la velocidad[3]. La ecuación usada para calcular la aceleración de cada robot del enjambre en el trabajo de *Shucker et al* es:

$$\mathbf{u}_i \quad = -k_d\dot{\mathbf{x}} + \sum_{j\in\mathcal{N}_j} k_s(|\mathbf{x}_i - \mathbf{x}_j| - I_0)\hat{\mathbf{v}}_{ji}$$

donde k_d y k_s son constantes del controlador, $\mathbf{x}_i$ y $\mathbf{x}_j$ son las posiciones de los robots i y j, I_0 es la distancia deseada a los robots, $\hat{\mathbf{v}}_{ji}$ es un vector unitario entre la posición del robot j y la del robot i, y $\mathcal{N}_i$ es el conjunto de robots en el entorno perceptual del robot i.

Una forma de usar este mecanismo de control en el modelo integrador (simple) es asignar las velocidades como:

$$\mathbf{v}_i \quad = \sum_{j\in\mathcal{N}_j} k_s(|\mathbf{x}_i - \mathbf{x}_j| - I_0)\hat{\mathbf{u}}_{ji}$$

donde el vector $\hat{\mathbf{v}}_{ji}$ se ha renombrado a $\hat{\mathbf{u}}_{ji}$ para que la notación no sea ambigua ($\mathbf{v}_i$ es la velocidad del robot i).

[2]Shucker et al. Convergence-Preserving Switching for Topology-Dependent Decentralized Systems. Trans. on Robotics, 2008

[3]El controlador del artículo es en realidad similar a un controlador PD.

2.3. Alejarse de Vecinos de otro Color

Para alejarse de los vecinos de otro color se puede implementar una fuerza/velocidad al estilo repulsivo de los Boids proporcional, por ejemplo, al inverso de la distancia. Esta componente de la velocidad final se debe combinar con las anteriores para generar el comportamiento deseado.

2.4. Función de evaluación

Como hemos visto el código a implementar tiene una serie de parámetros que afectarán al comportamiento de los enjambres. Aspectos como la distancia entre robots del enjambre, el número de grupos que se forman o el tamaño de cada grupo son algunas de las características que pueden depender de estos parámetros. Es importante destacar dos cosas. Primero, puesto que los robots se inicializan aleatoriamente habrá una componente aleatoria en los valores citados (número de grupos, distancias, etc) y, segundo, los valores dependerán del instante temporal en que se evaluen los enjambres (por ejemplo el número de grupos será menor según pase el tiempo ya que más y más robots se encontrarán y agruparán). Dada la aleatoriedad y la complejidad que implica anallizar el comportamiento emergente del enjambre es conveniente disponer de un método para evaluar dicho comportamiento que proporcione valores numéricos que midan cierta característica deseada de los enjambres. Para ello habrá que implementar una función de evaluación que genere dichos valores numéricos que pueden ser: número de enjambres de robots de cada color, tamaño de los enjambres existentes en términos del número de robots, dispersión espacial de cada grupo/enjambre (por ejemplo media de las distancias entre los robots de un grupo y su desviación estándar), o cualquier otra que mida una característica deseada para el comportamiento final del enjambre. Estas características se pueden evaluar después de cierto tiempo de simulación o a lo largo de toda la simulación para ver su evolución temporal. Dado que las inicializaciones de los robots son aleatorias es conveniente ejecutar varias evaluaciones de las características para un conjunto de parámetros.

3. Código a implementar

Para implementar los enjambres de robots que realicen movimiento coordinado utilizaremos el simulador simple ya visto en las sesiones de prácticas, aunque ha sido necesario añadir cierta funcionalidad para poder trabajar con ángulos y con la topología del entorno. El fichero `aligningRobot.hh` contiene la declaración de la clase a implementar. Es necesario implementar el método `action()` de la clase `AligningRobot` en el fichero `aligningRobot.cc`, así como la función `analyseSwarm()` del fichero `aligningSwarm.cc`. Esta última será la función que analice el comportamiento de los enjambres, y se ha incluido en la función `main()` código que llama a la función para la evolución temporal de los robots. Los resultados pueden guardarse en un fichero o imprimirse por pantalla dentro de la función `analyseSwarm()`. Para facilitar la implementación, en el fichero de cabecera `base.hh` se han incluido declaraciones de las siguientes funciones (implementadas en el fichero `environment.cc`):

- `Position2d average(const std::vector<Position2d>& );` Esta función calcula el punto medio de las posiciones pasadas como argumento en un vector de

entrada. La posición de retorno de la función es el punto medio en el entorno toroidal. Esta función puede utilizarse para implementar el comportamiento de cohexión en el enjambre. El punto medio relativo al robot también puede calcularse (sin usar esta función) como la media estándar de los vectores de posiciones relativas de los vecinos. Los vectores de posiciones relativas pueden obtenerse a través de la distancia y ángulo entre las posiciones de los robots en el entorno.

- `float averageAngle(const std::vector<float>& );` Esta función calcula el promedio de una serie de ángulos pasados como argumento de entrada en un vector estándar. Esta función puede utilizarse para implementar el comportamiento de alineación del enjambre para promediar las direcciones (ángulos) de las velocidades de los robots vecinos, aunque no es necesario utilizarla si se promedian directamente los vectores de velocidad.

- `float averageAngleW(const std::vector<float>& , const std::vector<float>& );` Esta función calcula el promedio de un conjunto de ángulos (primer argumento) pesados con ciertos pesos (segundo argumento). Puesto que el cálculo de una media angular (pesada) no tiene solución analítica, esta función implementa una optimización a través de una búsqueda en rejilla (*grid search*) que la hace muy poco eficiente. El valor de retorno es una aproximación a la media, y los pesos deben ser no negativos aunque no es necesario que sumen 1. Esta función se puede utilizar para implementar el comportamiento de separación en los robots del enjambre y/o para combinar los comportamientos de alineación, cohexión y separación si se opta por implementar Boids. Al igual que en los casos anteriores no es necesario utilizarla si se usan operaciones vectoriales, siempre que se tenga en cuenta cómo definir vectores de posición relativa en el entorno.

4. Estructura y Contenidos del Informe

La estensión máxima del informe será de seis páginas (incluyendo bibliografía, figuras, etc) con una fuente 10pt, 11pt o 12pt y margenes de 20mm o más, y se enviará exclusivamente en formato PDF. Se proporciona un fichero LaTeXcomo prototipo para el informe para quien quiera usarlo. Se recomienda que el informe del trabajo siga la estructura de un artículo científico también conocida como *IMMRaD* (*Introduction, Materials and Methods, Results and Discussion*)[4]. A continuación se describe lo que, idealmente, debe contener cada sección, aunque la inclusión o no de determinados aspectos lo decidirá cada autor/a.

- Título y autor: Se debe elegir un título representativo del trabajo y el nombre del autor/a debe incluirse en el documento.

- Resumen: Resumen del documento completo que debe tocar los siguientes aspectos de la estructura (Introducción, Métodos y Resultados, ver secciones específicas). El resumen debe ser lo último a escribir, ya que puede simplemente consistir en parafrasear texto del documento.

[4] Aunque los nombres de las secciones no tiene por que ser esos.

- Introducción: Idealmente la introducción debe incluir una descripción del problema, así como establecer su importancia. La introducción suele incluir una revisión de lo que se llama estado del arte (*State-of-the-art* o *S-o-t-A*), que consiste en una breve descripción de trabajos que hayan propuesto soluciones más o menos distintas al problema del que trata el documento. Lo ideal en trabajos científicos sería identificar limitaciones en las soluciones existentes y proponer soluciones a esas limitaciones. Aunque esto no es extrictamente necesario en este trabajo es un buen ejercicio intentar responder a la pregunta ¿Por qué merece la pena hacer esto?, i.e. justificar el trabajo y su importancia a cierto nivel.

- Materiales y Métodos: En este caso los materiales (el simulador usado) no son muy relevantes y es más apropiado concentrarse en los métodos. Esta sección debe explicar en detalle qué mecanismos se han usado para el trabajo. Los métodos usados deben explicarse a un nivel de abstracción alto, es decir, se pueden incluir ecuaciones matemáticas, algoritmos, pseudocódigo y descriciones textuales de los métodos pero no hay que incluir código en ningún lenguaje de programación (el código se entregará de forma separada). Por otro lado las descripciones y/o formulaciones matemáticas usadas deben tener suficiente detalle para que un lector con ciertos conocimientos pueda replicar el trabajo (si dispone de los materiales necesarios).

- Resultados: En esta sección se presentarán resultados de los experimentos propuestos. Los experimentos deben pensarse de antemano para probar o ilustrar algún elemento del proyecto. Es aconsejable definir una medida cuantitativa de lo que se quiere conseguir, es decir, uno o varios números cuyos valores teóricos o límites puedan predecirse y que alguno de ellos indique el nivel de consecución o éxito del método.

- Discusión: Idealmente esta sección puede ser una de las más complicadas de escribir, aunque por lo general se limita a un resumen del trabajo y los resultados principales. Idealmente los resultados deberían compararse con los trabajos previos presentados en el *S-o-t-A* para argumentar que las limitaciones identificadas al principio del informe se han superado en el trabajo. Este es también el lugar para exponer limitaciones de la implementación realizada y posibles extensiones del trabajo.

A. Average computation in manifolds (Cálculo de medias en variedades)

In this appendix we will see how the average of a set of points can be computed when the points belong to a manifold, i.e. a topological space locally similar to a Euclidean space. Manifolds appear often in fields like robotics or machine learning (e.g. manifold learning) so it is useful to understand somethings about them. Dealing with points in manifolds can be complicated as we will see in this appendix, that is why a typical assumption in many fields is that points (data or other) belong to an Euclidean space, which can lead to good enough results, yet not always. A prototypical example in robotics of a variable belonging to a manifold is an angular variable θ. Let's assume we have a set of angles $\theta_1, \theta_2, \cdots, \theta_n$ in the range $(-\pi, \pi]$ and we want to calculate their averare. As a first step let's assume we have only two angles $\theta_1 = -10^\circ$ and $\theta_1 = 10^\circ$. According to the standard definition of the mean[5] the average $\bar{\theta} = \frac{1}{2}(\theta_1 + \theta_2) = 0$, which makes sense. Now take $\theta_1 = -170^\circ$ and $\theta_1 = 170^\circ$. If we use the same formula to calculate the average we get to $\bar{\theta} = \frac{1}{2}(\theta_1 + \theta_2) = 0$, which clearly does not make sense since the average of both angles should be 180°. There is something wrong on the way we calculate the average in this case, and the problem is that the set of angles does not live in an Euclidean space[6]. This problem also appears if we consider, for instance, vectors in the unit circle or vectors with a given norm/length. Take two vectors in the unit circle $\hat{v}_1 = [1, 0]$ and $\hat{v}_2 = [0, 1]$, if we average them as usual the result is $v = \frac{1}{2}(v_1 + v_2) = [\frac{1}{2}, \frac{1}{2}]$ which does not belong to the unit circle since $|v| = \frac{1}{\sqrt{2}}$, and in fact its norm is smaller than 1.

To calculate the average of a set of points $x_1, x_2, \cdots, x_n$ in a manifold $\mathcal{M}$ such as the unit circle, i.e. average of angles, we need to find the point $\bar{x}$ solving the following minimisation problem:

$$\bar{x} = \underset{x \in \mathcal{M}}{\arg\min} \sum_{i=1}^{n} d(x_i, x) \tag{1}$$

where x_i $i = 1, 2, \cdots, n$ are the points, $\bar{x}$ is the average point which must belong to the manifold $\bar{x} \in \mathcal{M}$, $d(x, y)$ is a distance[7] between points x and y in the manifold, and $\arg\min_x()$ takes the value x which minimises the value of the sum. Equation (1) is the definition of the average which leads to the know equation for the mean. To see this let's take a set of points in a Euclidean space (e.g. the real line $\Re$) and the square of the Euclidean distance on the real line ($d(x, y) = (x - y)^2$). For a set of points $\{x_1, x_2, \cdots, x_n\}$ the average is therefore defined as:

$$\bar{x} = \underset{x \in \mathcal{M}}{\arg\min} \sum_{i=1}^{n} (x_i - x)^2 \tag{2}$$

[5] This actually derives from the definition of the average for Euclidean spaces using the square of the Euclidean distance as we will see sortly.

[6] Angular variables actually belong to the unit circle S^1 instead of the real line $\Re$.

[7] Remember that a distance $d(\cdot, \cdot)$ is a function of two inputs fulfilling: 1) $d(x, x) = 0$, $d(x, y) > 0$ and $d(x, y) = d(y, x)$.

The problem is now to find x which minimises $\sum_{i=1}^{n}(x_i - x)$ for the given points, but since equation (2) is simple enough we can obtain the average just by setting to zero the derivative of the sum w.r.t. x and try to solve for x, i.e.

$$\begin{aligned}\frac{d}{dx}\sum_{i=1}^{n}(x_i - x)^2 &= \sum_{i=1}^{n}\frac{d}{dx}(x_i - x)^2 \\ &= \sum_{i=1}^{n} -2(x_i - x) \\ &= -2\sum_{i=1}^{n} x_i + 2\sum_{i=1}^{n} x = -2\sum_{i=1}^{n} x_i + 2nx \end{aligned} \tag{3}$$

Setting (3) equal to zero and solving for x leads to the well known result:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4}$$

If we want to calculate the average of a set of angles we need to define a distance which, in this case, should account for the topology of the angular variables, i.e. take the shortest path between two angles. One such distances could look like[8]:

$$d(\alpha, \beta) = \begin{cases} \alpha - \beta & \text{if } |\alpha - \beta| < \pi \\ \alpha - \beta + 2\pi & \text{if } \alpha - \beta < -\pi \\ \alpha - \beta - 2\pi & \text{if } \alpha - \beta > \pi \end{cases} \tag{5}$$

But trying to use this distance in the definition of the average, eq (1) leads to a constrained optimisation problem with no analytic solution. However, it can be shown that for a set of n angles, the average can be obtained from measuring the sum of distances to the following "average candidates" and taking the one with the smaller distance:

$$\bar{\theta}_j = \frac{1}{n}\sum_{i=1}\theta_i + j\frac{2\pi}{n} \tag{6}$$

where $j = 1, 2 \cdots, n$.

[8] Actually this distance measures the length of the shortest path on the unit circle between two angles, since the arc lenght of an angle in the circle is $l = r\alpha$, with α in radians and $r = 1$.