FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

TA154 - Robótica Móvil

Práctica 1 - Transformaciones, locomoción y sensado

1C2025

Dominguez Lautaro Tadeo 109645 ltdominguez@fi.uba.ar

Índice

1. Transformaciones 2D y matrices afines

La pose de un robot en el plano, con respecto a una terna de referencia global (G), puede representarse como:

$$x = (x, y, \theta)^T \tag{1}$$

donde $(x, y)^T$ indica la posición del robot en el plano y (θ) su orientación.

Supongamos que el robot se encuentra en la pose:

$$x_1 = (x_1, y_1, \theta_1)^T \tag{2}$$

Desde esa ubicación, detecta un obstáculo (p) cuya posición relativa es (p_x, p_y), es decir, que está visto desde el sistema de referencia local del robot $(R1)$. Como se puede ver en la siguiente figura:

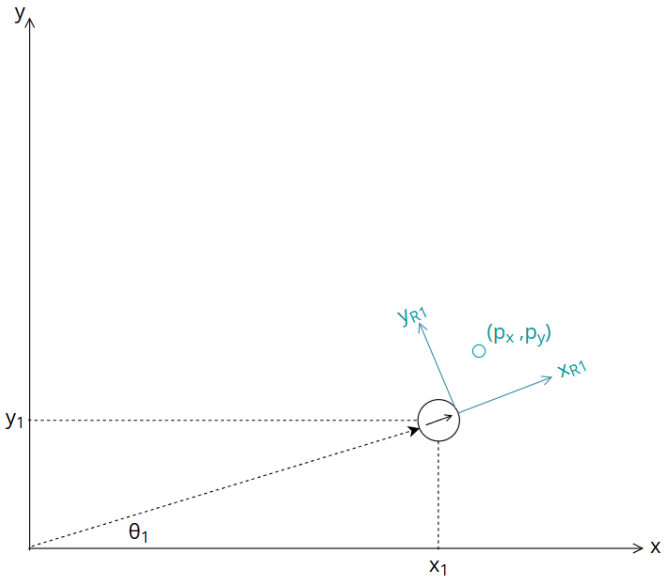

Figura 1: Pose 1

Se puede expresar la coordenada p con respecto a la terna global, de la siguiente forma:

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix}_G = T_{G \leftarrow R1} \cdot \begin{bmatrix} p_x \\ p_y \end{bmatrix}_{R1} \tag{3}$$

Donde la matriz $T_{G \leftarrow R1}$ está definida como:

$$T_{G \leftarrow R1} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

También es posible expresar una coordenada dada en el sistema de referencia global (G) en el sistema de referencia del robot $(R1)$, mediante la transformación $T_{R1 \leftarrow G}$, la cual corresponde a la inversa de la transformación $T_{G \leftarrow R1}$, es decir:

$$T_{R1 \leftarrow G} = T_{G \leftarrow R1}^{-1} \tag{5}$$

Resultando en la siguiente matriz:

$$T_{R1 \leftarrow G} = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & -x_1 \cdot \cos\theta_1 - y_1 \cdot \sin\theta_1 \\ -\sin\theta_1 & \cos\theta_1 & x_1 \cdot \sin\theta_1 - y_1 \cdot \cos\theta_1 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

El robot se desplaza a una nueva pose $x_2 = (x_2, y_2, \theta_2)^T$ en el sistema de referencia global, como se muestra en la figura.

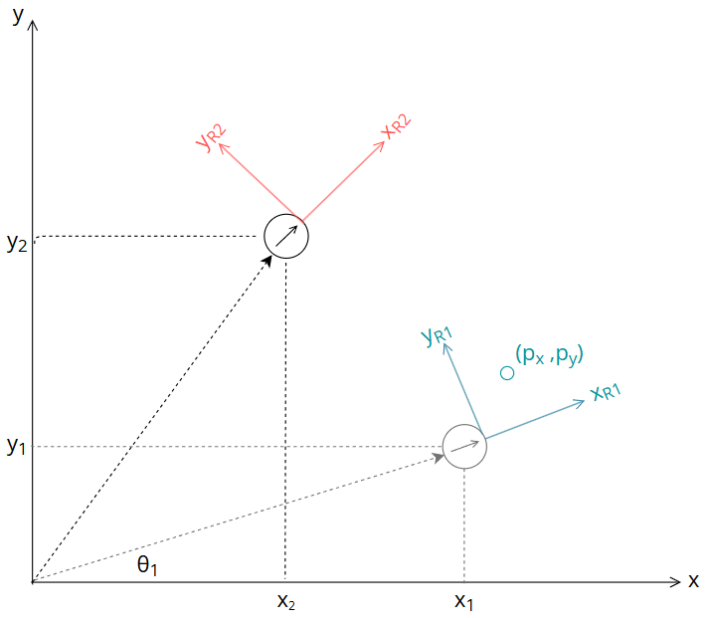

Figura 2: Pose 2

De la misma forma se puedo obtener $T_{G\leftarrow R2}$ y $T_{R2\leftarrow G}$

$$T_{G\leftarrow R2} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & x_2 \\ \sin\theta_2 & \cos\theta_2 & y_2 \\ 0 & 0 & 1 \end{bmatrix} \quad , \quad T_{R2\leftarrow G} = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & -x_2\cdot\cos\theta_2 - y_2\cdot\sin\theta_2 \\ -\sin\theta_2 & \cos\theta_2 & x_2\cdot\sin\theta_2 - y_2\cdot\cos\theta_2 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

Para obtener la matriz de transformación del sistema de referencia del robot en la pose 1 al sistema del robot en la pose 2, se utiliza la siguiente expresión:

$$T_{R2\leftarrow R1} = T_{R2\leftarrow G}\cdot T_{G\leftarrow R1} \tag{8}$$

Resultando en la siguiente matriz:

$$T_{R2\leftarrow R1} = \begin{pmatrix} \cos(\theta_1-\theta_2) & \sin(\theta_2-\theta_1) & x_1\cos(\theta_2)+y_1\sin(\theta_2)-x_2\cos(\theta_2)-y_2\sin(\theta_2) \\ \sin(\theta_1-\theta_2) & \cos(\theta_1-\theta_2) & y_1\cos(\theta_2)-x_1\sin(\theta_2)+x_2\sin(\theta_2)-y_2\cos(\theta_2) \\ 0 & 0 & 1 \end{pmatrix} \tag{9}$$

Esta matriz permite transformar un punto expresado en coordenadas relativas a la terna $R1$ hacia coordenadas relativas a la terna $R2$, de la siguiente manera.

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix}_{R2} = T_{R2\leftarrow R1}\cdot \begin{bmatrix} p_x \\ p_y \end{bmatrix}_{R1} \tag{10}$$

2. Sensado

En este trabajo se analiza el entorno percibido por un robot móvil que cuenta con un sensor LIDAR montado sobre su estructura. El objetivo es representar las mediciones del sensor tanto en su propio sistema de referencia como en la terna global, utilizando transformaciones homogéneas.

El robot se encuentra ubicado en una posición global definida por:

$$(x_{robot}, y_{robot}, \theta_{robot})^T_{Global} = (5, -7, -\frac{\pi}{4})^T \tag{11}$$

Sobre el robot, está montado un sensor LIDAR cuya posición y orientación están definidas respecto al cuerpo del robot como:

$$(x_{lidiar}, y_{lidiar}, \theta_{lidiar})^T_{Robot} = (0{,}2, 0, \pi)^T \tag{12}$$

Que al transformar las mediciones a la terna global, su pose es:

$$(x_{lidiar}, y_{lidiar})^T_{Global} = (5{,}14, -7{,}14)^T \tag{13}$$

Este sensor realiza un barrido angular de apertura total π, comenzando desde $-\frac{\pi}{2}$ hasta $\frac{\pi}{2}$, tomando muestras a intervalos angulares uniformes. Cada medición representa la distancia desde el sensor hasta un punto del entorno. Estos datos se encuentran en el archivo *laserscan.dat*, lo cuales fueron proyectados sobre el sistema de referencia del LIDAR para obtener la siguiente gráfica representativa de los puntos del entorno tal como los ve el sensor, sin considerar aún la posición ni la orientación del robot:

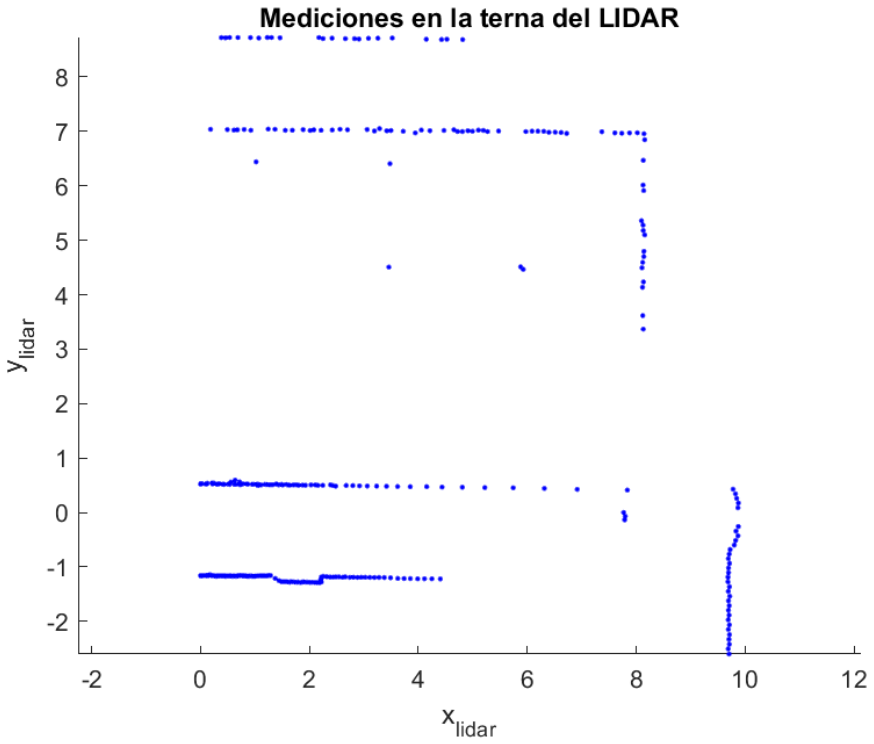

Figura 3: Mediciones en la terna del LIDAR

En la siguente figura, se muestran las mediciones transformadas al sistema de referencia global, utilizando transformaciones homogéneas. Además, se incluye la ubicación del robot ,marcada con una cruz roja.

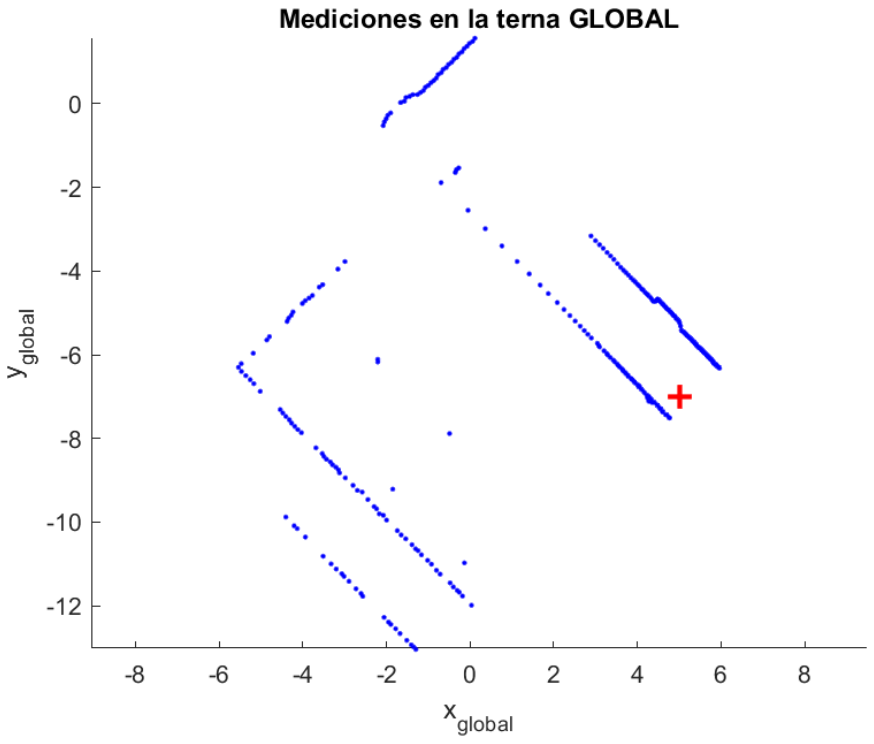

Figura 4: Mediciones en la terna del Global

Las mediciones obtenidas permiten reconstruir una imagen parcial del entorno en el que se encuentra el robot. Cada punto representa la distancia medida por el LIDAR en una dirección específica dentro de su campo visual. En la gráfica, se observa una distribución de puntos que podría corresponder a paredes, pasillos, equinas u otros elementos.

Las figuras y coordenadas presentadas en esta sección fueron obtenidas a través del siguiente script desarrollado en MATLAB:

```
% Parametros de la pose del robot en la terna global (G)
theta_robot_G = -pi/4;
x_robot_G = 5;
y_robot_G = -7;

% Parametros de la pose del LIDAR respecto al robot (R)
theta_lidar_R = pi;
x_lidar_R = 0.2;
y_lidar_R = 0;

% Matriz de transformacion del robot en la terna global
T_Global_Robot = [cos(theta_robot_G), -sin(theta_robot_G), x_robot_G;
                  sin(theta_robot_G),  cos(theta_robot_G), y_robot_G;
                  0, 0, 1];

% Matriz de transformacion del LIDAR en la terna del robot
T_Robot_Lidar = [cos(theta_lidar_R), -sin(theta_lidar_R), x_lidar_R;
                 sin(theta_lidar_R),  cos(theta_lidar_R), y_lidar_R;
                 0, 0, 1];

% Posicion del LIDAR en la terna global
p_lidar_G = T_Global_Robot * [x_lidar_R; y_lidar_R; 1]
x_lidar_G = p_lidar_G(1);
y_lidar_G = p_lidar_G(2);

% Matriz de transformacion directa del LIDAR a la terna global
T_Global_Lidar = T_Global_Robot * T_Robot_Lidar;

% Cargar datos del LIDAR (vector de distancias)
scan = load('-ascii','laserscan.dat');

% Calcular angulos correspondientes a cada rayo LIDAR
angles = linspace(-pi/2, pi/2, length(scan));

% Puntos del LIDAR en su propia terna (coordenadas homogeneas)
scan_lidar = [scan .* cos(angles);
              scan .* sin(angles);
              ones(1, length(scan))];

% Transformar las mediciones a la terna global
scan_global = T_Global_Lidar * scan_lidar;

% Graficar en la terna del LIDAR
figure; hold on; axis equal;
plot(scan_lidar(1, :), scan_lidar(2, :), 'b.');
title('Mediciones en la terna del LIDAR');
xlabel('x_{lidar}');
ylabel('y_{lidar}');

% Graficar en la terna GLOBAL
figure; hold on; axis equal;
plot(scan_global(1, :), scan_global(2, :), 'b.');
plot(x_robot_G, y_robot_G, 'r+', 'MarkerSize', 10, 'LineWidth', 2); %Posicion del robot
title('Mediciones en la terna GLOBAL');
xlabel('x_{global}');
ylabel('y_{global}');
```

Listing 1: Script Sensado

3. Accionamiento diferencial

3. Accionamiento diferencial

En esta sección se implementa la cinemática para un robot móvil de accionamiento diferencial, es decir, un robot que se desplaza gracias al movimiento independiente de sus dos ruedas laterales. Este modelo cinemático permite calcular la nueva posición y orientación del robot a partir de su pose inicial y las velocidades impuestas a las ruedas durante un intervalo de tiempo determinado.

Para ello, se desarrolló en `MATLAB/Octave` la fución que se muestra a continuación, que toma como entrada la pose actual del robot (x, y, θ), las velocidades de las ruedas izquierda y derecha (v_l, v_r), la duración del movimiento t, la distancia entre ruedas l, y el radio de las ruedas r. Como salida, la función devuelve la nueva pose del robot después de ejecutar dicho movimiento.

```
function [x_n, y_n, theta_n] = diffdrive(x, y, theta, v_l, v_r, t, l)
    % Calculo de velocidades en la terna del robot
    v = (1/2)*(v_l + v_r);          % Velocidad lineal del robot
    w = (1/l)*(v_r - v_l);          % Velocidad angular del robot

    % Vector de velocidades en la terna del robot
    v_robot = [v; 0; w];

    % Matriz de transformacion del robot a la terna global
    T_Global_Robot = [cos(theta), -sin(theta), 0;
                      sin(theta),  cos(theta), 0;
                      0,           0,          1];

    % Velocidad en la terna global
    v_global = T_Global_Robot * v_robot;

    % Nueva pose luego de aplicar durante t segundos
    x_n = x + v_global(1) * t;
    y_n = y + v_global(2) * t;
    theta_n = theta + v_global(3) * t;
end
```

Listing 2: Funcion de accionamineto diferencial

A partir de esta función, se simularon distintas acciones de control que el robot ejecuta en secuencia, partiendo desde una pose inicial dada. Cada acción está definida por un par de velocidades para las ruedas y una duración. Finalmente, se representa gráficamente la trayectoria recorrida por el robot a lo largo de toda la secuencia, lo que permite analizar su comportamiento y validar el funcionamiento del modelo.

A continuación, se muestra el código que utiliza la función `diffdrive` para simular una secuencia de acciones de control. En este script, se define la pose inicial del robot, un conjunto de acciones (pares de velocidades para las ruedas y duración), y se almacena la trayectoria resultante a lo largo del tiempo. Luego, dicha trayectoria es graficada:

```
% Parametros iniciales
x = 0; y = 0; theta = pi/4;
l = 0.5;
dt = 0.1; % paso de simulaci n mas peque o

% Secuencia de acciones: [v_l, v_r, t]
acciones = [
    0.1   0.5   2;
    0.5   0.1   2;
    0.2   0.2   2;
    1.0   0.0   4;
    0.4   0.4   2;
    0.2  -0.2   2;
    0.5   0.5   2
];

% Almacenar trayectoria
trayectoria = [x; y];

```

```
% Ejecutar cada accion en pasos de dt
for i = 1:size(acciones, 1)
    v_l = acciones(i,1);
    v_r = acciones(i,2);
    t_total = acciones(i,3);

    % Dividir la acci0n en peque os pasos de tiempo
    for t = 0:dt:(t_total-dt)
        [x, y, theta] = diffdrive(x, y, theta, v_l, v_r, dt, l);
        trayectoria(:, end+1) = [x; y];
    end
end

% Graficar trayectoria
figure; hold on; axis equal;
plot(trayectoria(1,:), trayectoria(2,:), 'b-');
plot(trayectoria(1,1), trayectoria(2,1), 'go', 'MarkerSize', 8, 'DisplayName', 'Inicio');
plot(trayectoria(1,end), trayectoria(2,end), 'ro', 'MarkerSize', 8, 'DisplayName', 'Fin');
xlabel('x [m]'); ylabel('y [m]');
title('Trayectoria del robot diferencial');
legend;
```

Listing 3: Funcion de accionamineto diferencial

La figura a continuación muestra la trayectoria completa del robot diferencial resultante de la ejecución secuencial de las acciones definidas:

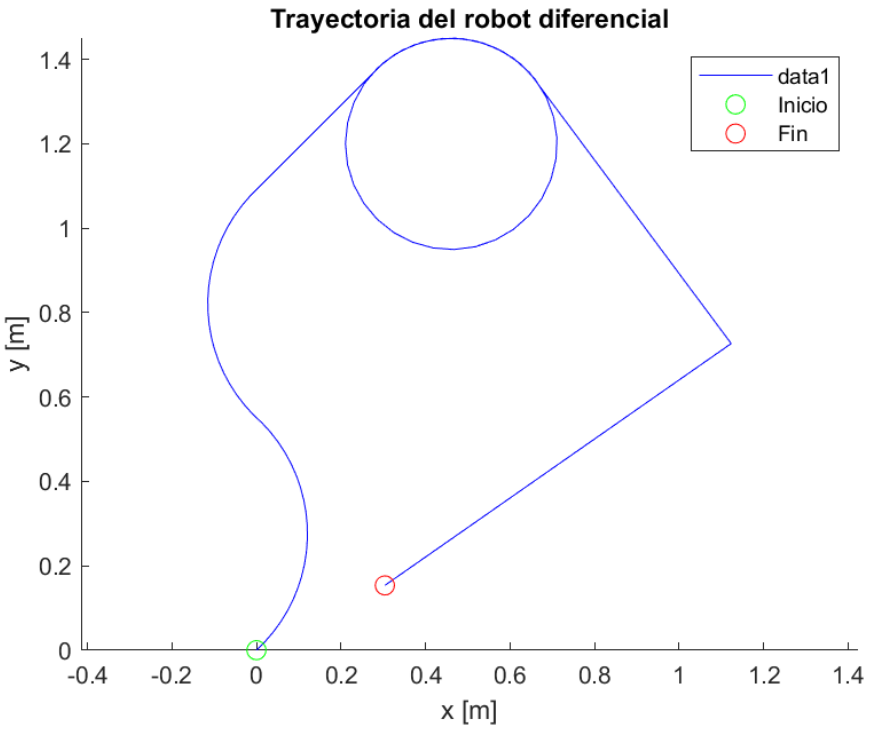

Figura 5: Trayectoria del robot