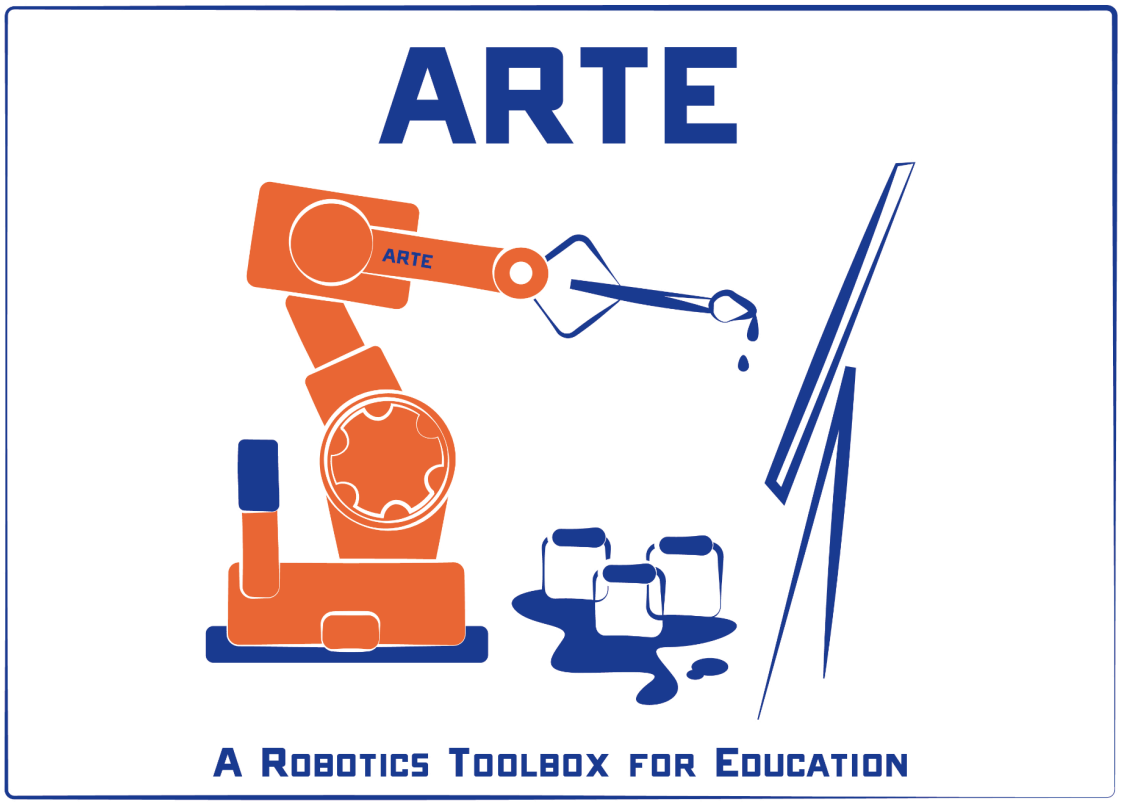


SESIÓN PRÁCTICA 3: DINÁMICA INVERSA

Arturo Gil Aparicio

arturo.gil@umh.es

OBJETIVOS

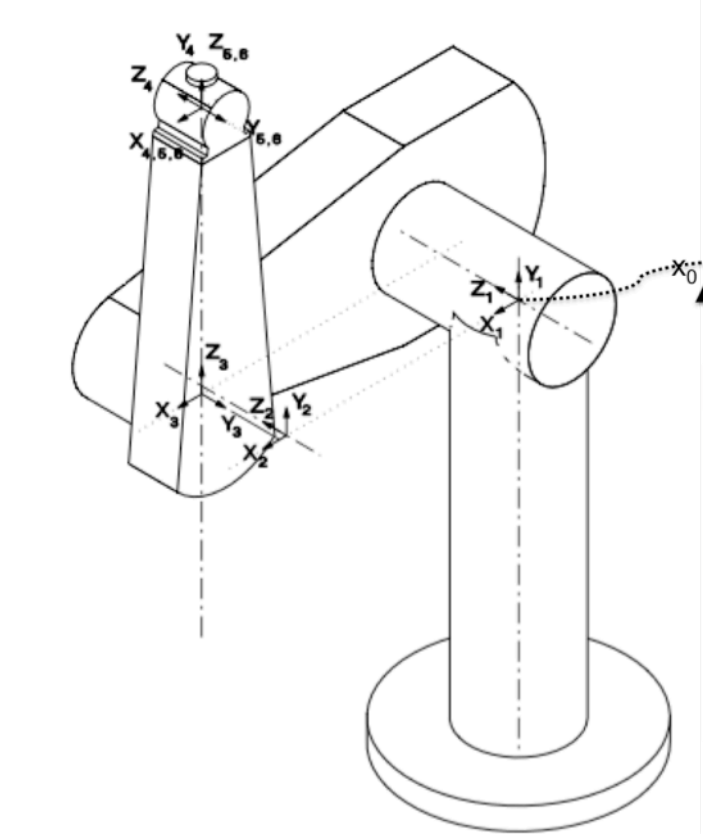
Se trabajará sobre los siguientes conceptos en esta práctica:

- Cálculo de la dinámica inversa de un manipulador usando el algoritmo de Newton-Euler.
- Cálculo de los pares/fuerzas en diferentes situaciones. Obtención del mejor y peor caso.
- Selección de motores basándonos en perfiles trapezoidales de aceleración.
- Simulación del sistema motor/robot y primeros pasos hacia el concepto de control.

Índice

1 Conceptos iniciales

El análisis dinámico de un mecanismo es un concepto avanzado. Puede considerarse dentro del área de conocimiento de Robótica aunque es un concepto amplio que engloba una gran cantidad de técnicas en las áreas de Física, Mecánica e Ingeniería. En muchas ocasiones los parámetros dinámicos de un robot no son proporcionados por el fabricante del mismo. Pondremos, por tanto, como ejemplo, el caso de un robot que ha sido estudiado de forma extensiva en el pasado por muchos investigadores: el robot UNIMATE PUMA 560. En la Figura 1 se presenta la tabla de parámetros DH del robot.



	Theta	D	A	Alfa
1	q_1	0	0	π/2
2	q_2	0	0.4318	0
3	q_3	0.15005	0.0203	-π/2
4	q_4	0.4318	0	π/2
5	q_5	0	0	-π/2
6	q_6	0	0	0

Figure 1

Los parámetros dinámicos de este robot se encuentran en la librería ARTE bajo la carpeta arte/robots/unimate/puma560/parameters.m. Observe los parámetros y comente con el profesor de prácticas si tiene alguna duda sobre ellos.

```
function robot = parameters()
%KINEMATICS
robot.name= 'puma_560';

robot.DH.theta = '[q(1) q(2) q(3) q(4) q(5) q(6)]';
robot.DH.d='[0  0   0.15005  0.4318     0    0.04]';
robot.DH.a='[0 0.4318  0.0203      0       0    0]';
robot.DH.alpha= '[pi/2  0   -pi/2  pi/2   -pi/2 0]';
robot.J=[];
robot.name='Puma 560 robotic arm';

robot.inversekinematic_fn = 'inversekinematic_puma560(robot,
T)';


%R: rotational, T: translational
robot.kind=['R' 'R' 'R' 'R' 'R' 'R'];

%number of degrees of freedom
robot.DOF = 6;

%minimum and maximum rotation angle in rad
robot.maxangle =[deg2rad(-160) deg2rad(160); %Axis 1, minimum,
maximum
                 deg2rad(-110) deg2rad(110); %Axis 2, minimum,
maximum
```

```
                deg2rad(-135) deg2rad(135); %Axis 3
                deg2rad(-266) deg2rad(266); %Axis 4: Unlimited
                deg2rad(-100) deg2rad(100); %Axis 5
                deg2rad(-266) deg2rad(266)]; %Axis 6: Unlimited

%maximum absolute speed of each joint rad/s or m/s
robot.velmax = [1
                1
                1
                1
                1
                1];%not available
% end effectors maximum velocity
robot.linear_velmax = 0.5; %m/s, from datasheet
robot.accelmax=robot.velmax/0.1;% 0.1 is here an acceleration
time

%base reference system
robot.T0 = eye(4);

%INITIALIZATION OF VARIABLES REQUIRED FOR THE SIMULATION
%position, velocity and acceleration
robot=init_sim_variables(robot);
robot.path = pwd;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GRAPHICS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%read graphics files
robot.graphical.has_graphics=1;
robot.graphical.color = [150 180 130]./255;
%for transparency
robot.graphical.draw_transparent=0;
%draw DH systems
robot.graphical.draw_axes=1;
%DH system length and Font size, standard is 1/10. Select 2/20,
3/30 for
%bigger robots
robot.graphical.axes_scale=1;
%adjust for a default view of the robot
robot.axis=[-1 1 -1 1 -0.66 2];
robot = read_graphics(robot);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%DYNAMIC PARAMETERS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
robot.has_dynamics=1;

%consider friction in the computations
robot.dynamics.friction=0;

%link masses (kg)
robot.dynamics.masses=[0 17.4 4.8 0.82 0.34 0.09];

%COM of each link with respect to own reference system
robot.dynamics.r_com=[0       0          0; %(rx, ry, rz) link 1
    -0.3638  0.006   0.2275; %(rx, ry, rz) link 2
    -0.0203 -0.0141  0.070;  %(rx, ry, rz) link 3
    0       0.019       0;%(rx, ry, rz) link 4
```

```
    0       0            0;%(rx, ry, rz) link 5
   0         0         -0.008];%(rx, ry, rz) link 6

%Inertia matrices of each link with respect to its D-H reference
system.
% Ixx   Iyy Izz Ixy Iyz Ixz, for each row
robot.dynamics.Inertia=[0      0.35 0       0   0   0;
    0.13    0.524   0.539   0   0   0;
    0.066   0.086   0.0125  0   0   0;
    1.8e-3  1.3e-3  1.8e-3  0   0   0;
    0.3e-3  0.4e-3  0.3e-3  0   0   0;
    0.15e-3 0.15e-3 0.04e-3 0   0   0];


%Please note that we are simulating the motors as presented in
MAXON
%catalog
robot.motors=load_motors([5 5 5 5 5 5]);

%Actuator rotor inertia
%Speed reductor at each joint. Please note that, for simplicity
in control, we consider that the gearratios are all positive
robot.motors.G=[62.6111 107.815 53.7063 76.0364 71.923 76.686];
```

Si llamamos a la función `load_robot`, cargaremos todos estos parámetros en una variable del entorno de trabajo (workspace) de Matlab. La variable `robot.has_dynamics` indica en la librería que existen parámetros dinámicos para el robot. Con los comandos siguientes cargamos el robot, calculamos una cinemática directa y lo dibujamos en la librería.

```
>> init_lib
>> robot=load_robot('unimate','puma560')
>> T=directkinematic(robot, [0 0 0 0 0 0])

T =

    1.0000         0         0    0.8000
         0   -1.0000   -0.0000   -0.0000
         0    0.0000   -1.0000    0.4000
         0         0         0    1.0000
>> drawrobot3d(robot, [0 0 0 0 0 0])
```

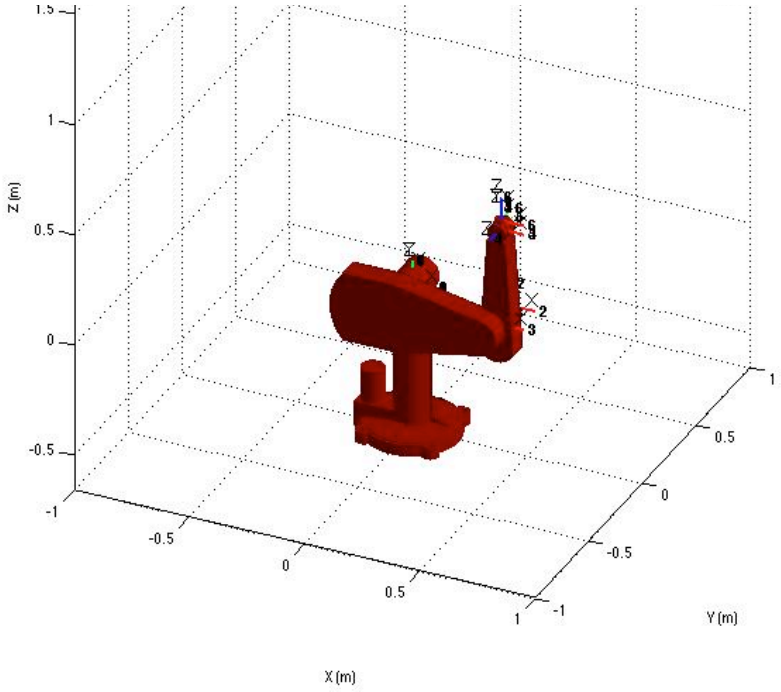


Figure 2

2 Dinámica inversa

Supuesto que, en un momento determinado, conocemos la posición articular, velocidad articular y aceleraciones en un brazo, podemos calcular los pares que instantáneamente llevarían al robot a ese estado cinemático. Así pues, podemos analizar las fuerzas y pares en las articulaciones en diferentes estados de funcionamiento del robot. Para observar esto en el robot PUMA UNIMATE podemos ejecutar los siguientes comandos

```
>> init_lib
>> robot=load_robot('unimate','puma560')
>> q1 = [0 0 0 0 0 0];
>> drawrobot3d(robot, q1)
>> tau = inversedynamic(robot, q1, [0 0 0 0 0 0], [0 0 0 0 0 0], [0  0
9.81]', [0 0 0 0 0 0]')
```

La función de dinámica inversa toma la posición articular q, velocidad articular vq y aceleración articular aq. Los últimos parámetros indican el vector de aceleración gravitatoria g y las fuerzas y momentos externos sobre el efector final del robot.

```
tau = inversedynamic(robot, q1, vq, aq, [0  0 9.81]', [0 0 0 0 0 0]')
```

Pruebe con diferentes valores de posición articular q, velocidad articular y aceleración articular.

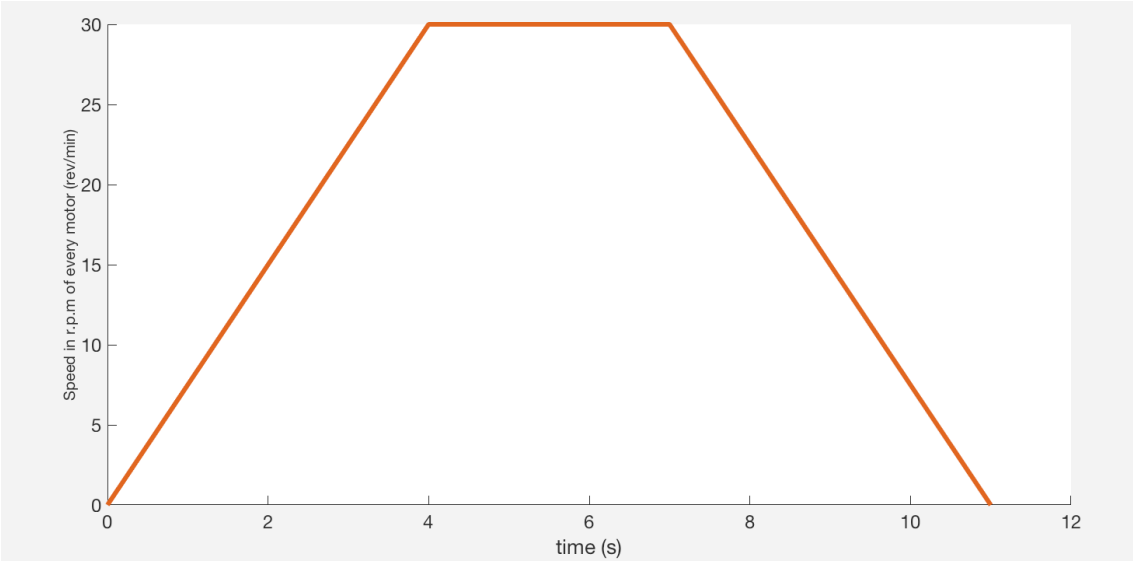
Ejercicio:

a) Busque la peor posición inicial del robot (caso más solicitado).

b) Busque la mejor posición inicial del robot (caso menos solicitado).

3 Selección de motores: creación de un script

Los robots industriales suelen tener, por defecto, una aceleración de tipo trapezoidal en sus articulaciones. Esto quiere decir, que cada articulación deberá moverse siguiendo un perfil de velocidad parecido al siguiente:



Observamos tres tramos:

- aceleración.
- velocidad constante.
- deceleración.

Abra en este momento el script

`arte/practicals/session3b/motor_selection_2dofplanar.m`.

Los parámetros siguientes rigen la forma de este perfil de aceleración. En concreto, la velocidad máxima de la articulación (no del motor) y la aceleración máxima de cada articulación.

```
maximum_speeds=[pi pi];%rad/second
maximum_accels=[pi/1 pi/1]; %rad/second^2
time_at_constant_speed=3; %seconds
```

Ejercicio:

El estudiante deberá modificar los valores anteriores para obtener unas especificaciones cinemáticas que le parezcan correctas para la aplicación del robot.

Se propone al estudiante que complete el script anterior para calcular los pares en las articulaciones para todo el movimiento trapezoidal anterior. Para simplificar el cálculo de la dinámica inversa y la búsqueda de motores nos vamos a centrar en un robot de 2GDL plano. El estudiante deberá completar el script anterior buscando los siguientes objetivos:

- El objetivo que se persigue es encontrar los pares nominales y pico en cada articulación.
- Al cambiar la variable `robot.motors.G = [1 1]` también podremos calcular (aproximadamente y sin considerar el rozamiento) los pares (nominales y pico) en cada articulación.

Tenga en cuenta que ya se le proporciona al estudiante una función que crea un perfil trapezoidal con un vector de velocidades y aceleraciones articulares. La posición articular debe considerarse fija al valor inicial.

```
[input_speeds, input_accels,
time]=build_trapezoidal_speed_profile(maximum_speeds, maximum_accels,
total_time);
```

Ejercicio:

Complete la función siguiente que permite calcular los pares en todas las articulaciones del robot para la posición, velocidad y aceleraciones dadas en el tiempo.

```
compute_inverse_dynamics(q, input_speeds, input_accels, time);
```

Requisitos:
- La función deberá plotear los pares en cada articulación como función del tiempo.

- La función deberá plotear los pares en cada motor como función del tiempo.

- La función deberá plotear la velocidad en rpm de cada motor como función del tiempo.

- La función deberá hallar, para cada articulación:
    - El par pico.
    - El par nominal.
    - La velocidad máxima de cada motor.

4 Selección de motores: Búsqueda de motores

Con el script anterior funcionando, deberíamos ser capaces de realizar los siguientes pasos:

1. Simular los pares en el robot/motores variando robot.motors.G = [1 1]
2. Para cada simulación deberíamos tener un resultado por pantalla como el siguiente:

```
MAIN RESULTS (referred to each motor):
-----------------------------------------------------------------------------
                              Joint 1 - Joint 2
Peak Torque (Nm):          63.193     18.121
Nominal Torque (Nm):       41.202     10.791
Max motor speed (r.p.m.): 30.0         30.0
-----------------------------------------------------------------------------
```

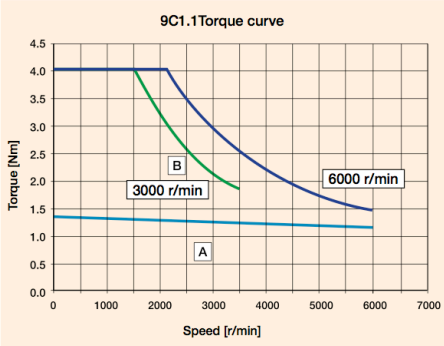
3. Note el estudiante que con esos tres valores (Par pico, par nominal y velocidad máxima) podemos buscar un motor en cualquier catálogo de motores. Se proporciona al estudiante un catálogo de motores ABB que puede consultar (ABB_catalog). Se proporcionan más catálogos para que el estudiante pueda observar otros fabricantes y sus especificaciones.

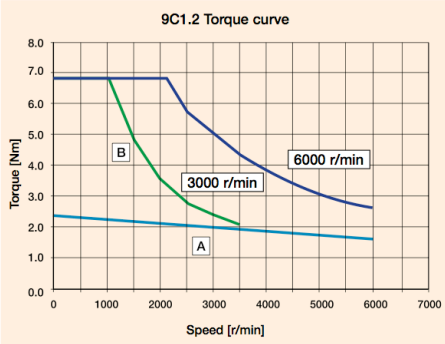
9C series technical details

Type	Continuous torque at zero speed [5] T_{cs} [Nm]	Current at continuous torque [1) 3) 5)] I_{cs} [A]	Rated torque [5)] T_{rat} [Nm]	Rated current [1) 3) 5)] I_{rat} [A]	Rated speed n_{rat} [r/min]	Rated frequency f_{rat} [Hz]	Mechanical rated power [5)] P_{rat} [kW]	Peak torque T_{pk} [Nm]	Current at peak torque [1) 3)] I_{pk} [A]	Torque constant [1) 2) 3)] k_T [Nm/A]	B.e.m.f. between phases at rated speed [1) 2) 3)] V [V]	Moment of inertia of rotor [3)] J_M [kgcm²]	Moment of inertia of rotor + brake [3)] J_M with brake [kgcm²]	Weight [3) 4)] W [kg]
9C1.1.30.. ..M	1.4	1.3	1.3	1.4	3000	250.0	0.41	4.1	4.5	1.147	208	0.57	0.62	3.0
9C1.2.30.. ..M	2.3	1.8	2	1.7	3000	250.0	0.63	6.9	6.1	1.440	261	1.04	1.09	4.0
9C1.3.30.. ..M	3.2	2.7	2.8	2.5	3000	250.0	0.88	9.6	9.0	1.350	245	1.51	1.56	5.0
9C1.4.30.. ..M	4.2	3.3	3.5	2.9	3000	250.0	1.10	12.6	11.1	1.440	261	1.99	2.04	6.0
9C1.1.60.. ..M	1.4	2.1	1.2	2.0	6000	500.0	0.75	4.1	7.1	0.720	261	0.57	0.62	3.0
9C1.2.60.. ..M	2.3	3.6	1.6	2.7	6000	500.0	1.01	6.9	12.1	0.720	261	1.04	1.09	4.0
9C1.3.60.. ..M	3.2	5.2	2.3	3.9	6000	500.0	1.45	9.6	17.3	0.702	255	1.51	1.56	5.0
9C1.4.60.. ..M	4.2	6.5	2.5	4.1	6000	500.0	1.57	12.6	21.6	0.738	268	1.99	2.04	6.0
9C4.1.30.. ..M	4.3	3.0	3.9	2.8	3000	250.0	1.23	12.9	9.8	1.654	300	4.0	4.7	5.6
9C4.2.30.. ..M	7.5	5.0	6.1	4.3	3000	250.0	1.92	22.5	16.7	1.704	309	7.6	8.3	7.9
9C4.3.30.. ..M	9.4	6.0	6.9	4.6	3000	250.0	2.17	28.2	19.9	1.786	324	11.1	11.8	10.2
9C4.4.30.. ..M	12.0	8.2	7.5	5.4	3000	250.0	2.36	36.0	27.3	1.665	302	14.7	15.4	12.5
9C4.1.40.. ..M	4.3	4.0	3.7	3.6	4000	333.3	1.55	12.9	13.2	1.232	298	4.0	4.7	5.6
9C4.2.40.. ..M	7.5	6.9	5.4	5.2	4000	333.3	2.26	22.5	23.1	1.232	298	7.6	8.3	7.9
9C4.3.40.. ..M	9.4	7.8	5.8	5.1	4000	333.3	2.43	28.2	26.1	1.365	330	11.1	11.8	10.2
9C4.4.40.. ..M	12.0	10.0	6.3	5.5	4000	333.3	2.64	36.0	33.3	1.365	330	14.7	15.4	12.5
9C5.2.20.. ..M	12.3	6.1	10.3	5.3	2000	166.7	2.16	36.9	20.2	2.307	279.0	21.8	23.6	15.5
9C5.3.20.. ..M	18.4	9.2	14.8	7.8	2000	166.7	3.10	55.2	30.7	2.272	274.7	31.6	33.4	19.2
9C5.4.20.. ..M	23.5	11.9	17.1	9.1	2000	166.7	3.58	70.5	39.6	2.249	272.0	41.4	43.2	22.9
9C5.5.20.. ..M	26.0	12.0	20.0	9.8	2000	166.7	4.19	78.0	40.2	2.452	296.5	51.2	53.0	26.6
9C5.6.20.. ..M	30.0	13.1	22.0	10.1	2000	166.7	4.61	90.0	43.8	2.596	313.9	61.0	62.8	30.3
9C5.2.30.. ..M	12.3	9.2	9.0	7.1	3000	250.0	2.83	36.9	30.8	1.515	274.7	21.8	23.6	15.5
9C5.3.30.. ..M	18.4	12.4	12.4	8.8	3000	250.0	3.90	55.2	41.3	1.688	306.1	31.6	33.4	19.2
9C5.4.30.. ..M	23.5	15.4	14.0	9.7	3000	250.0	4.40	70.5	51.4	1.731	313.9	41.4	43.2	22.9
9C5.5.30.. ..M	26.0	17.1	17.0	11.8	3000	250.0	5.34	78.0	56.9	1.731	313.9	51.2	53.0	26.6
9C5.6.30.. ..M	30.0	19.7	18.0	12.4	3000	250.0	5.65	90.0	65.7	1.731	313.9	61.0	62.8	30.3

4. Busque dos motores que cumplan con estos requisitos. Fíjese en la masa y tamaño del motor. Si no lo considera adecuado, cambie G y repita el proceso. Es importante que compruebe que el par pico, par

nominal y velocidad máxima se encuentran dentro de la zona de trabajo en el diagrama de par/velocidad de cada motor. Note que, en general, hay una zona de trabajo intermitente (pico) y otra zona de trabajo nominal (continua).





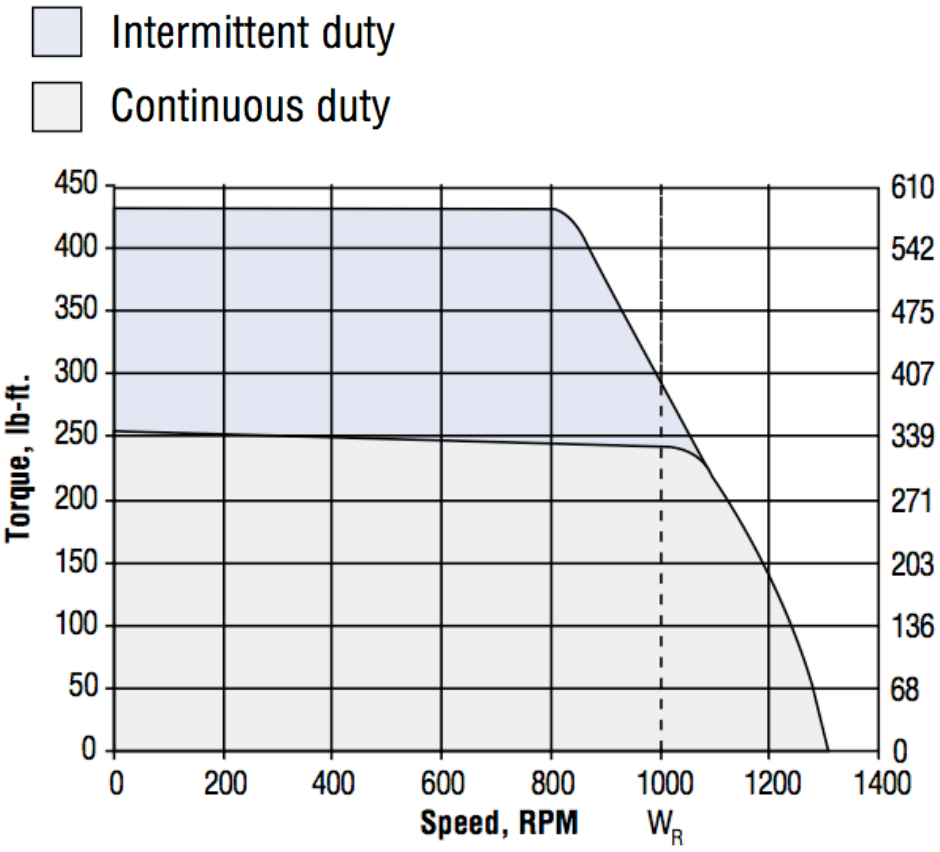


Figure 10

Tenga en cuenta que, según la convención seguida por ARTE, G > 1 indica que el motor realiza un par menor que aquél correspondiente a la articulación. En contrapartida, el motor gira a una velocidad mayor que la de la articulación.

Ejercicio:

Seleccione dos motores del catálogo de ABB que se adecúen al robot de 2 GDL plano.

4 Prueba y simulación de los motores

Podemos, a continuación, probar los motores. Obviamente, podríamos comprar los motores, las reductoras, construir el robot y probarlo todo junto. No obstante, es preferible realizar una simulación previamente, de manera que podamos validar el conjunto previamente. Se plantea, por tanto, a continuación, la prueba en simulación del conjunto robot-motores. El alumno puede utilizar los motores elegidos en el apartado anterior. También, si lo desea, puede utilizar los datos correspondientes a estos motores y reductores.

Articulación 1: G= 20, motor ABB 9C1.4.30

Articulación 2: G= 10 motor ABB 9C1.3.30

Cada motorreductor tiene las siguientes características (del catálogo):

Articulación 1: G= 20, motor ABB 9C1.4.30
- Peak current: 11.1 A
- Torque constant: 1.44 Nm/A
- Voltage constant (261V girando a 3000 rpm): 0.087 V/rpm = 0.8307 V/rad/s
- R = 12 ohms
- L = 0.0165 H

Articulación 2: G= 10 motor ABB 9C1.3.30
- Peak current: 9.0 A
- Torque constant: 1.35 Nm/A
- Voltage constant (245V girando a 3000 rpm): 0.087 V/rpm = 0.7798 V/rad/s
- R = 12 ohms
- L = 0.0165 H

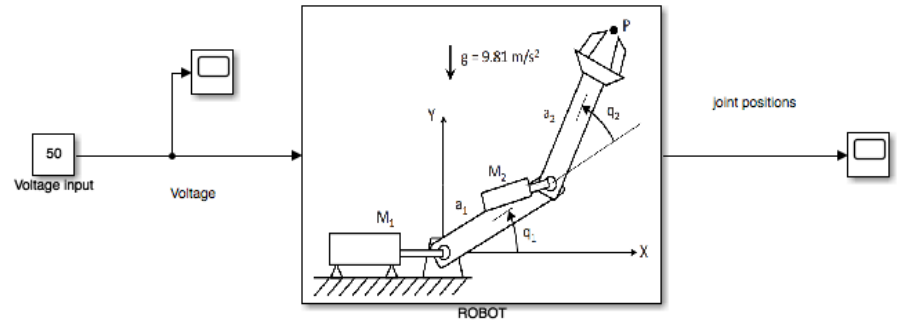
Se plantea realizar una simulación mínima. En esta simulación se desea comprobar si "al menos" los motores pueden mover el brazo en su peor posición (para la posición en la que fueron seleccionados los motores). En estos momentos no se desea, todavía, plantear un sistema de control, sino introducir un voltaje fijo a los motores y comprobar si estos mueven el robot. Realice los siguientes pasos:

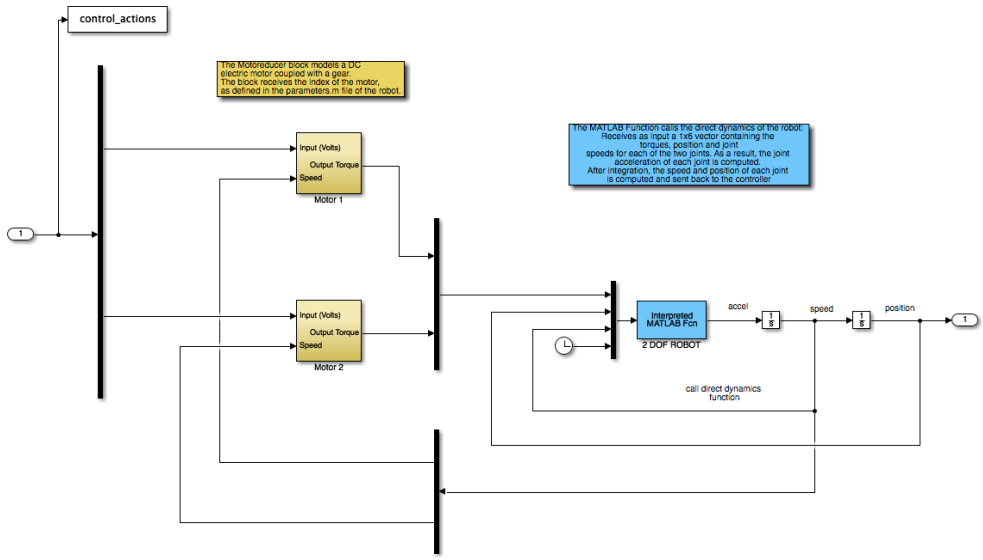
a) Cargue el robot e indique la reducción G.

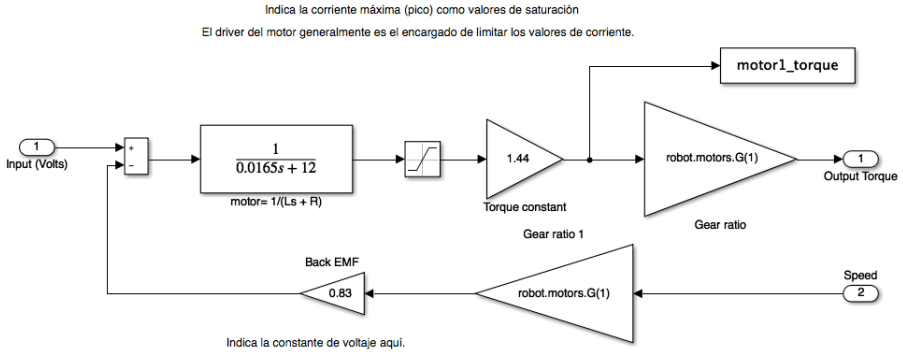
```
>> robot=load_robot('example', '2dofplanar');
>> robot.motors.G = [20 10];
```

puede variar estos valores en cualquier momento y ver cómo afectan a la simulación.

b) Cargue el esquema de Simulink simulate_robot_and_motors.slx. Se presentan algunas capturas de este modelo.







c) Navegue y familiarícese con el esquema.

- Busque dónde se encuentran las constante de cada motor del robot.
- Busque dónde se encuentra la entrada de voltaje de los motores.
- Busque cuál es la salida de los motores y qué tipo de variable es.
- Encuentre la función de dinámica directa.

c) Navegue y entienda la función de cada bloque. Obtenga un concepto global de la función total del sistema de simulación.

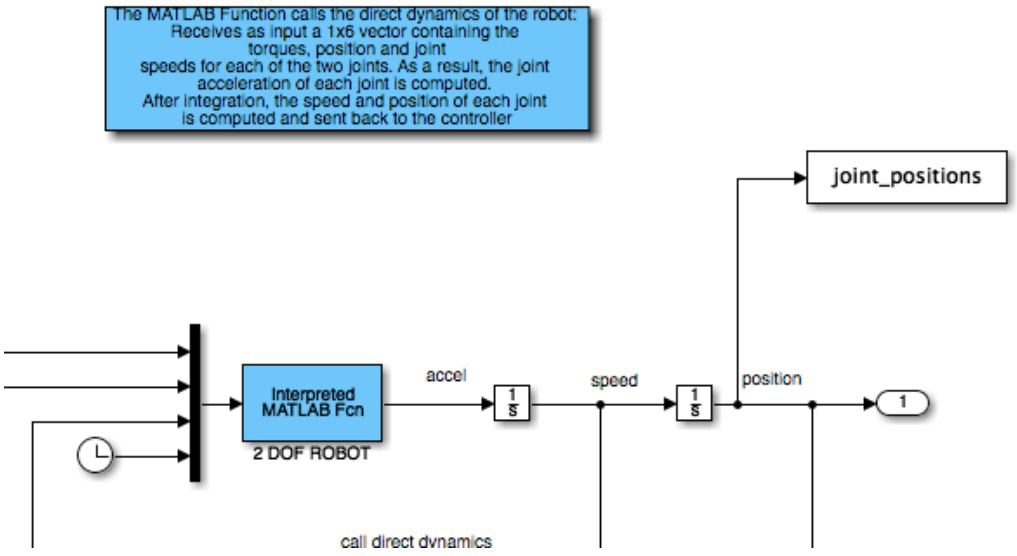
Realice las siguientes cuestiones:

a) Introduzca el valor de las constantes de los motores. Considere R, L, Kp, Kv y G. No olvide indicar la corriente máxima (bloque de saturación). Considere también dónde se encuentra el factor de reducción G.

b) Simule el sistema y comience a observar los resultados obtenidos. No olvide que el objetivo principal es considerar si el robot se va a mover o no, es decir: si los motores seleccionados son capaces de mover al robot.

c) Añada las siguientes variables al esquema. Utilice un bloque To Workspace para ello.

- joint_positions: posiciones articulares
- joint_torques: pares articulares
- joint_speeds: velocidades articulares
- joint_accelerations: aceleraciones articulares
- motor1_torque: par en el motor 1.
- motor2_torque: par en el motor 2.
- motor1_current: corriente en el motor 1.
- motor2_current: corriente en el motor 2.



Después de cada simulación, puede observar las variables anteriores con los siguientes comandos:

```
>> plot(joint_positions), legend('q1', 'q2')
>> plot(joint_torques), legend('tau_1', 'tau_2')
>> animate(robot, joint_positions')
```

c) Varie la entrada de voltaje, anote qué observa en cada caso:

Simule el sistema y comience a observar los resultados obtenidos. No olvide que el objetivo principal es considerar si el robot se va a mover o no, es decir: si los motores seleccionados son capaces de mover al robot. Simule para los siguientes valores de voltaje de entrada.

V = 0
V = 10
V = 20
V = 50

Anote sus propias conclusiones.

d) Imagine, ahora, que debe plantear un sistema de control para que se siga una trayectoria en posición y velocidad... qué variaciones realizaría sobre el anterior esquema para conseguir esto.