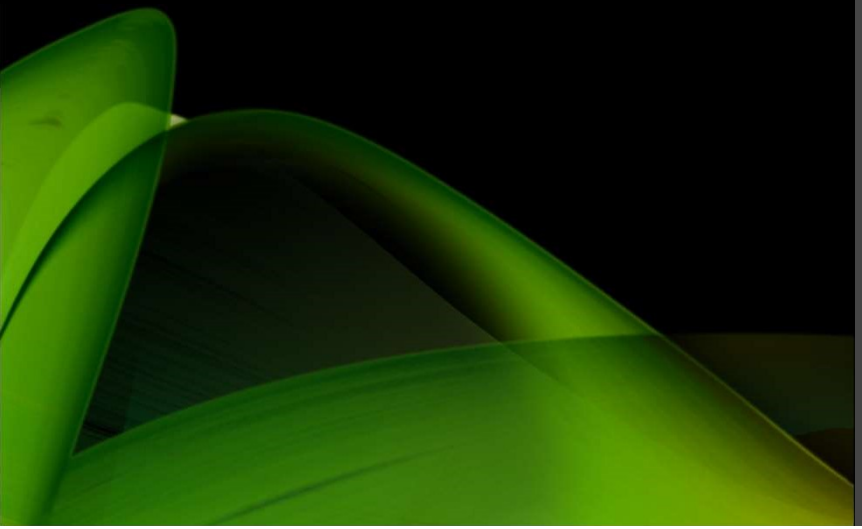
Agulló Soto, Rubén
Bonmatí Campello, Raúl
Sánchez Martí, Joaquín

SISTEMA AMBIDEXTRO DE MANIPULACIÓN ROBÓTICA

ÍNDICE:

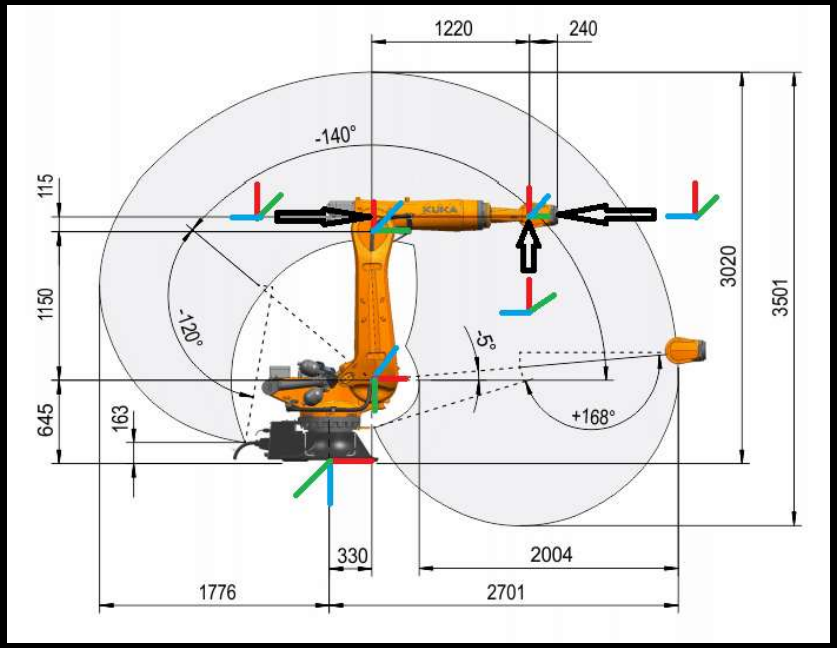


- Objetivo general
- Desarrollo implementado mediante MatLab (librería ARTE)
- Resultados y simulaciones
- Conclusiones

KUKA KR300_R2700_2

→ Cinemática directa:

-Disposición de los sistemas de referencia (siguiendo los planos del fabricante) en las distintas articulaciones del robot siguiendo el sistema de colores RGB(xyz), para posteriormente poder trasladarnos de un sistema de referencia a otro mediante los parámetros Denavit Hartenberg:



	θ	d	a	alpha
0_{A^1}	q(1)	-0,625	0,330	$\pi/2$
1_{A^2}	q(2) - $\pi/2$	1,150	0	0
2_{A^3}	q(3)	0,115	0	$\pi/2$
3_{A^4}	q(4)	0	1,220	- $\pi/2$
4_{A^5}	q(5)	0	0	$\pi/2$
5_{A^6}	q(6)	0	0,240	0

```
robot.name= 'KR300_R2700_2';

robot.DH.theta= '[q(1) q(2)-pi/2 q(3) q(4) q(5) q(6)]'
robot.DH.d='[-0.645 1.150 0.115 0 0 0]';
robot.DH.a='[0.330 0 0 1.220 0 0.240]';
robot.DH.alpha= '[pi/2 0 pi/2 -pi/2 pi/2 0]';
```

KUKA KR300_K2700_2

→ Cinemática inversa:

OBJETIVO GENERAL

- -Emular el funcionamiento de los brazos del robot ATLAS de Boston Dynamics para que estos sean capaces de transportar una caja.

- -Para esto haremos uso de la dinámica inversa, dinámica directa y path planning.

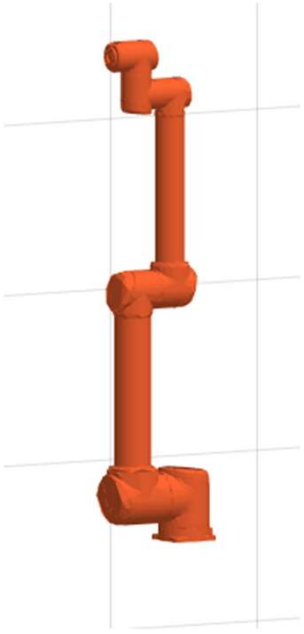
- -Emplearemos dos robots de Universal Robots llamados UR10.

Technical specifications UR10 — Item no. 1

We accept no liability for any printing errors or technic

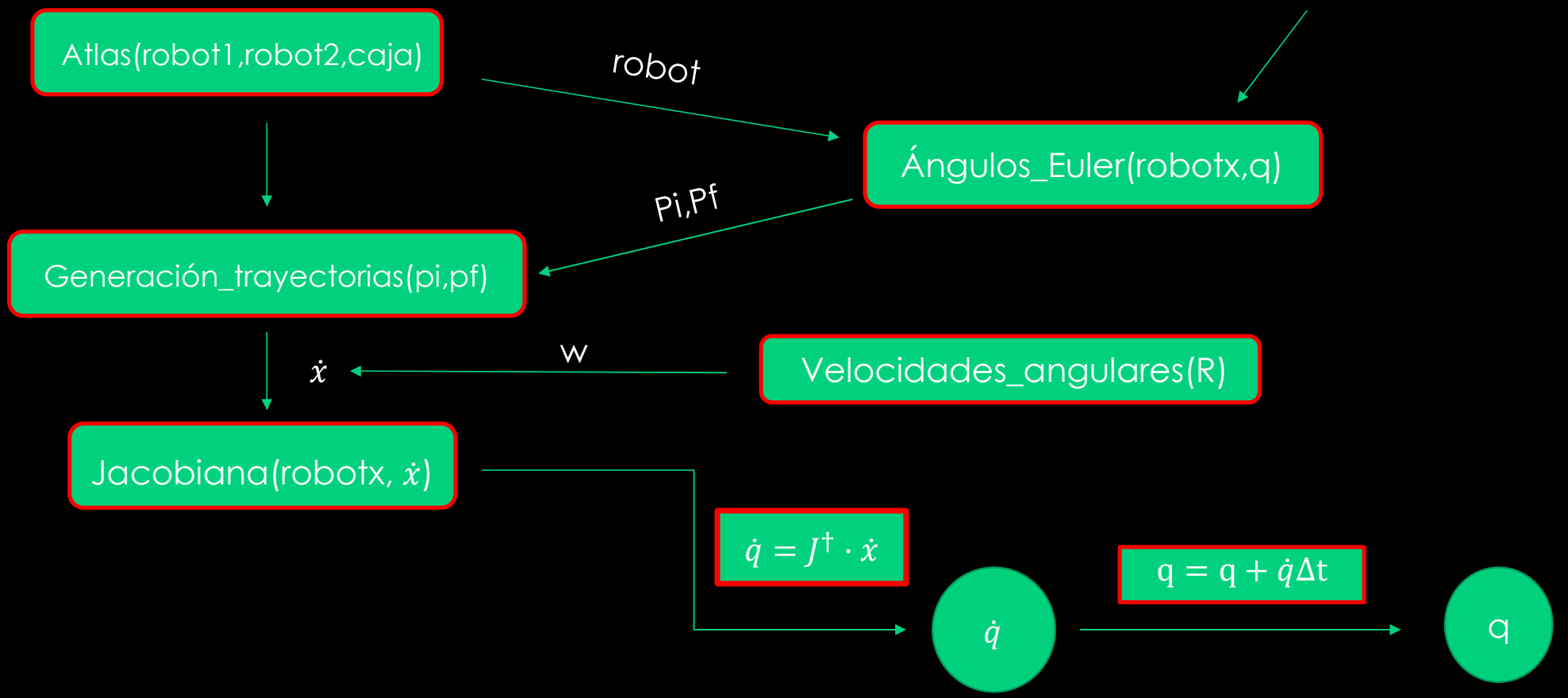
6-axis robot arm with a working radius of 1300 mm / 51.2 in

Weight:	28.9 kg / 63.7 lb
Payload:	10 kg / 22 lbs
Reach:	1300 mm / 51.2 in
Joint ranges:	+/- 360°
Speed:	Base and Shoulder: 120°/s. Elbow, Wrist 1, Wrist 2, Wrist 3: 180°/s. Tool: Typical 1 m/s. / 39.4 in/s.
Repeatability:	+/- 0.1 mm / +/- 0.0039 in (4 mils)
Footprint:	Ø190 mm / 7.5 in
Degrees of freedom:	6 rotating joints



ROBOT UR10

DESARROLLO


Q inicial
Q final
Atlas(robot1,robot2,caja)
robot
Ángulos_Euler(robotx,q)
Pi,Pf
Generación_trayectorias(pi,pf)
$\dot{x}$
w
Velocidades_angulares(R)
Jacobiana(robotx, $\dot{x}$)
$\dot{q} = J^{\dagger} \cdot \dot{x}$
$\dot{q}$
$q = q + \dot{q}\Delta t$
q

\PRACTICA transversal\atlas.m

```
function atlas(robot1,robot2,caja)
t = 1;
[pi]=angulos_euler(robot1,[-0.132 -0.967 -1.289 0.279 -0.021 0]);
[pi2]=angulos_euler(robot1,[0.1440 0.9641 1.3023 -0.0121 0.0544 -1.0145]);
[pf]=angulos_euler(robot1,[-0.119 0.405 -1.626 -0.74 0 2.011]);
[pf2]=angulos_euler(robot1,[0.1498 -0.4106 1.6997 0.8706 0.0273 -2.0933]);

xpunto1=generacion_trayectorias(pi,pf);
q1=jacobiana(robot1,xpunto1)

xpunto2=generacion_trayectorias(pi2,pf2);
q2=jacobiana(robot2,xpunto2);
q2=-q2;

while t<99
    drawrobot3d(robot1,q1(t,:)); % dibuja el primer robot
    hold on
    drawrobot3d(robot2,q2(t,:),1); % dibuja el segundo robot
    %----------dibuja la caja----------
    T=directkinematic(robot1,q1(t,:));
    caja.T0=T;
    caja.T0(1:3,1:3)=[1 0 0;0 0 -1;0 1 0];
    caja.T0(3,4)=caja.T0(3,4)-0.05;
    caja.T0(2,4)=caja.T0(2,4)-0.1;
    drawrobot3d(caja,zeros(1,6),1);
    %-----------------------------------
    pause(0.05);
    hold off
    t=t+1;
end
end
```

\PRACTICA transversal\angulos_euler.m

```
1   function [p]=angulos_euler(robot,q)
2
3 -   T=directkinematic(robot,q);
4 -   betha=asin(T(1,3));
5 -   gamma=asin(-T(1,2)/cos(betha));
6 -   alpha=acos(T(3,3)/cos(betha));
7 -   p=[T(1,4) T(2,4) T(3,4) alpha betha gamma];
8
9 -   end
```

$$\begin{bmatrix} \cos(\beta)\cdot\cos(\gamma) & -\cos(\beta)\cdot\sin(\gamma) & \sin(\beta) \\ \cos(\alpha)\cdot\sin(\gamma)+\sin(\alpha)\cdot\sin(\beta)\cdot\cos(\gamma) & \cos(\alpha)\cdot\cos(\gamma)-\sin(\alpha)\cdot\sin(\beta)\cdot\sin(\gamma) & -\sin(\alpha)\cdot\cos(\beta) \\ \sin(\alpha)\cdot\sin(\gamma)-\cos(\alpha)\cdot\sin(\beta)\cdot\cos(\gamma) & \cos(\alpha)\cdot\sin(\beta)\cdot\sin(\gamma)+\sin(\alpha)\cdot\cos(\gamma) & \cos(\alpha)\cdot\cos(\beta) \end{bmatrix}$$

\PRACTICA transversal\generacion_trayectorias.m

```
function [Xpunto]=generacion_trayectorias(pi,pf)
    i = 1;
    j=1;
    m=1;
    lambda = 0;
    dlambda = 0.01;
    while lambda<1
        p(i,:) = pi + (pf-pi)*lambda;%Interpolacion lineal de los puntos
        lambda = lambda + dlambda;
        rx=[1 0 0;0 cos(p(i,4)) -sin(p(i,4));0 sin(p(i,4)) cos(p(i,4))];
        ry=[cos(p(i,5)) 0 sin(p(i,5));0 1 0;-sin(p(i,5)) 0 cos(p(i,5))];
        rz=[cos(p(i,6)) -sin(p(i,6)) 0; sin(p(i,6)) cos(p(i,6)) 0;0 0 1];
        R{i}=rx*ry*rz; %matriz de matrices de rotación
        i = i + 1;
    end
    vx=diff(p(:,1)); %velocidad lineal=derivada de los vectores de posicion
    vy=diff(p(:,2));
    vz=diff(p(:,3));
    w=velocidades_angulares(R);%Scrip de las velocidades angulares
    while j<99
        wx(j)=w{j}(3,2);
        wy(j)=w{j}(1,3);
        wz(j)=w{j}(2,1);
        j=j+1;
    end
    %Montamos el vector Xpunto
    while m<99
        Xpunto(:,m)=[vx(m) vy(m) vz(m) wx(m) wy(m) wz(m)].';
        m=m+1;
    end
end
```

\PRACTICA transversal\jacobiana.m

```
1   function [qq]=jacobiana(robot,Xpunto)
2       i=1;
3       q = [-0.1440 -0.9810 -1.2830 0.0070 -0.0545 1.0050]';
4       pause on;
5
6       while i<99
7          J = compute_jacobian(robot,q);
8          qd = pinv(J)*Xpunto(:,i);
9          q = q + qd;
10
11         qq(i,:)=q; %almacenamos en una matriz todas las 'q'
12
13         i = i + 1;
14      end
15  end
```

$$\dot{q} = J^{\dagger} \cdot \dot{x}$$

\PRACTICA transversal\velocidades_angulares.m

```
function [w]=velocidades_angulares(R)
    j=1;
    m=1;
    b=1;

    %Desmontamos la matriz
    while j<100
        r11(j)=R{j}(1,1);
        r12(j)=R{j}(1,2);
        r13(j)=R{j}(1,3);

        r21(j)=R{j}(2,1);
        r22(j)=R{j}(2,2);
        r23(j)=R{j}(2,3);

        r31(j)=R{j}(3,1);
        r32(j)=R{j}(3,2);
        r33(j)=R{j}(3,3);

        j=j+1;
    end

    %Derivamos los vectores
    rd11=diff(r11);
    rd12=diff(r12);
    rd13=diff(r13);

    rd21=diff(r21);
    rd22=diff(r22);
    rd23=diff(r23);

    rd31=diff(r31);
    rd32=diff(r32);
    rd33=diff(r33);

    %Motamos la matriz derivada
    while m<99
        dR{m}(1,1)=rd11(m);
        dR{m}(1,2)=rd12(m);
        dR{m}(1,3)=rd13(m);

        dR{m}(2,1)=rd21(m);
        dR{m}(2,2)=rd22(m);
        dR{m}(2,3)=rd23(m);

        dR{m}(3,1)=rd31(m);
        dR{m}(3,2)=rd32(m);
        dR{m}(3,3)=rd33(m);

        m=m+1;
    end

    %Multiplicamos dR por R(transpuesta) para sacar w
    while b<99
        w{b}=dR{b}*(R{b}.');

        b=b+1;
    end
end
```

CONCLUSIONES

PROBLEMÁTICAS:

- -Muñeca no esférica :
- -Calcular la inversa → métodos numéricos
- -Interpolación usando Euler

- -Puntos singulares

- -Mostrar dos robots y la caja

CONCLUSIONES

Proyectos futuros:

-Trayectorias no lineales de la caja

-Mantener la caja con la base paralela al suelo

-Evitar colisiones

} Posibles mejoras

-Brazos de robot humanoide

-Robot colaborativo

-Transporte y paletizado

} Usos