Vision and Control of Industrial Robots 01211433

Lecture 11: Vision-based Control

Dr.Varodom Toochinda

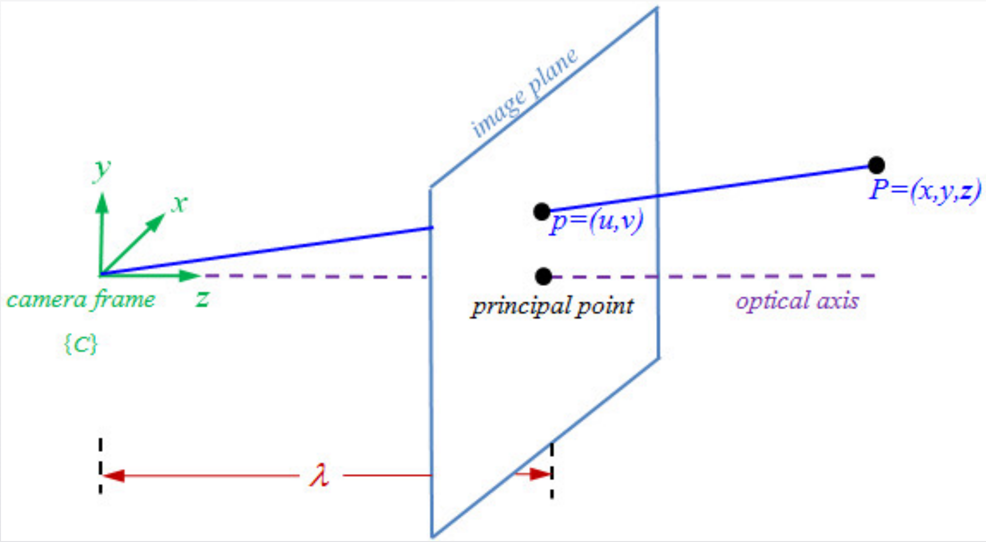
Dept. of Mechanical Engineering

Kasetsart University

Outline

- Vision-based control basics
- Image Processing
- Camera models
- Image features
- Vision-based control
- Image Jacobian matrix

Camera coordinated frame



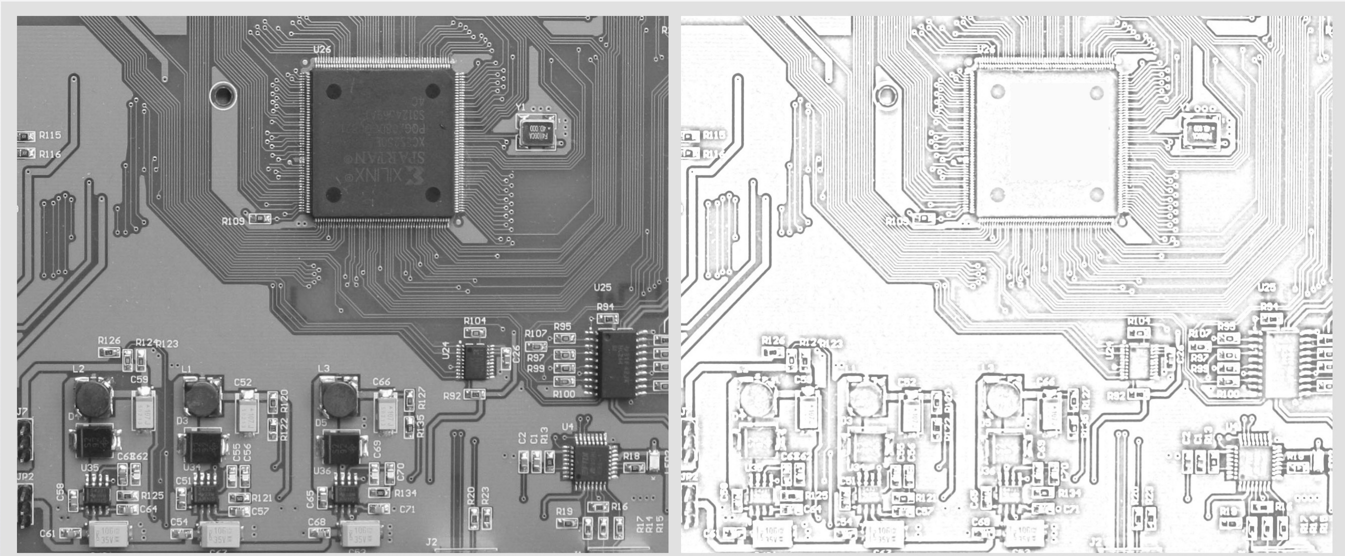
Perspective projection

$$k\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u \\ v \\ \lambda \end{bmatrix}$$

$$u = \lambda \frac{x}{z}, \quad v = \lambda \frac{y}{z}$$

Effects of perspective projection

Image enhancement and processing



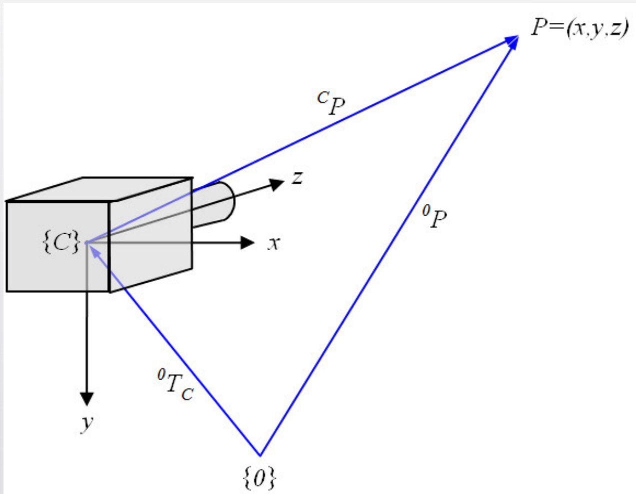
Camera model in RTSX

```
cam=CentralCamera('focal',0.015);
P = [0.4, 0.5, 3.0]';
CamProject(cam, P)
```

```
 ans  =
     0.002
     0.0025
```

Camera frame

```
CamProject(cam, P, 'Tcam', transl([-0.6, 0, 0]))

ans  =
    0.005
    0.0025
```



Camera matrix

```
cam = CentralCamera('focal', 0.015, 'pixel', 10e-6,'resolution',[1280 1024], 'center',[640 512], 'name', 'my camera');

CamProject(cam, P)

 ans  =
     840.
     762.
```

$$p = \underbrace{\left(\begin{bmatrix} \frac{\lambda}{s_x} & 0 & o_r \\ 0 & \frac{\lambda}{s_y} & o_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}^0T_C^{-1} \right)}_{C} P = CP$$

$$-\frac{u}{s_x} = (r - o_r), \quad -\frac{v}{s_y} = (c - o_c)$$

```
cam.C

ans  =
    1500.    0.       640.    0.
    0.       1500.    512.    0.
    0.       0.       1.      0.
```

Image Feature

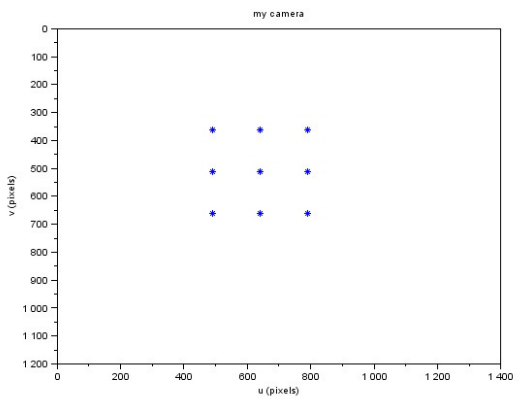
```
P = mkgrid(3, 0.2, 'T', transl([0,0,1.0]))
 P  =
  - 0.1  - 0.1  - 0.1    0.     0.    0.     0.1    0.1    0.1
  - 0.1    0.     0.1  - 0.1    0.    0.1  - 0.1    0.     0.1
    1.     1.     1.     1.     1.    1.     1.     1.     1.
```

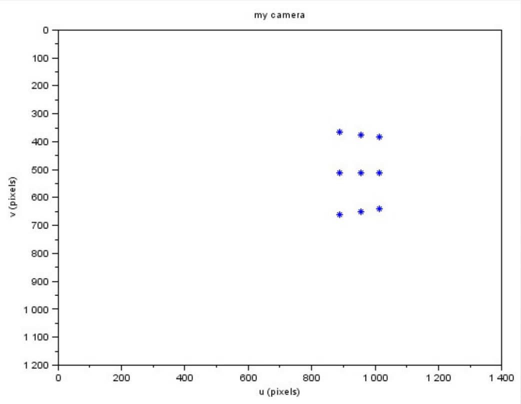
```
CamProject(cam,P)
 ans  =
   490.    490.    490.    640.    640.    640.    790.    790.    790.
   362.    512.    662.    362.    512.    662.    362.    512.    662.
```

```
CamPlot(cam, P);
```

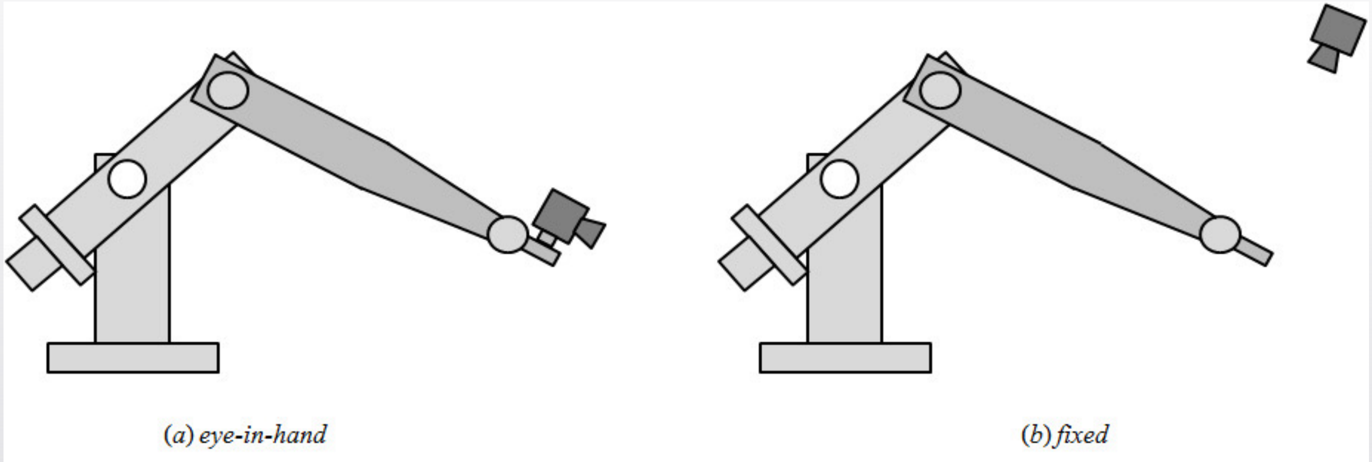
Image Feature

```
Tcam = transl([-1,0,0.5])*troty(0.9);

CamPlot(cam, P, 'Tcam', Tcam);
```





Vision-based control



Two common camera installation

Vision-based control configuration

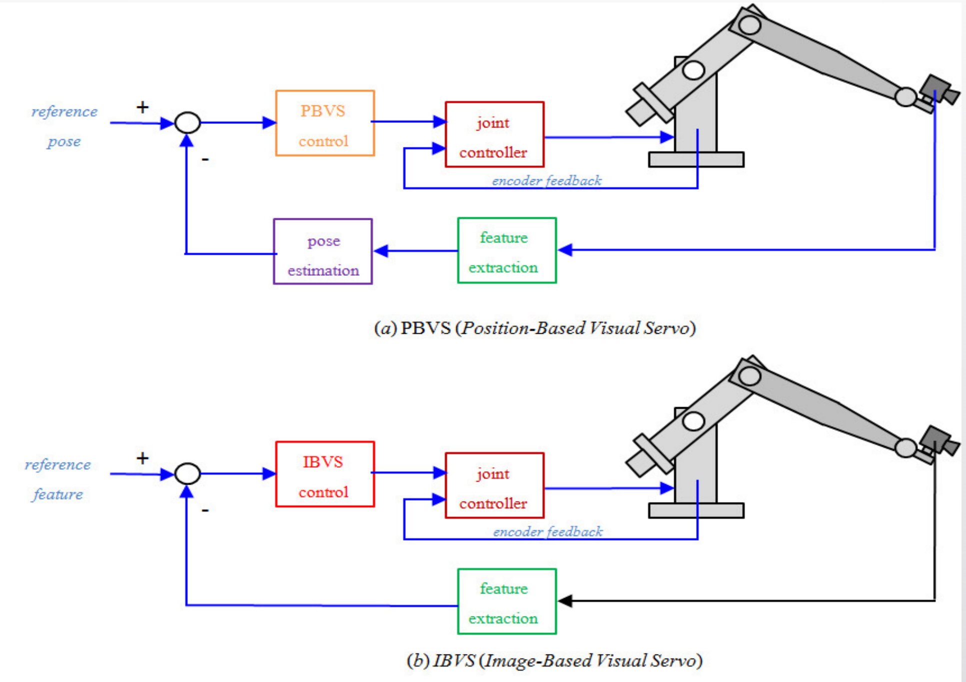
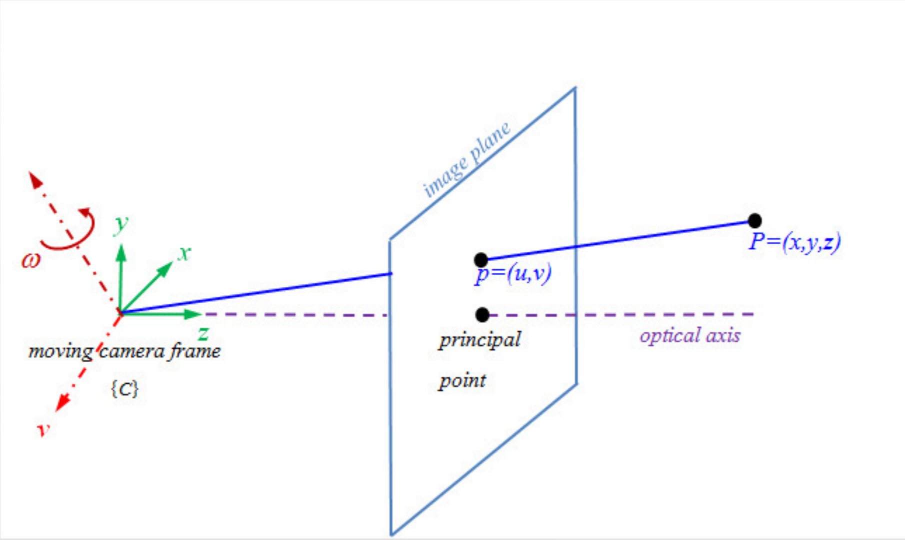


Image Jacobian matrix

$$\dot{s} = J_p(s, q)\xi$$

$$s(t) = \begin{bmatrix} u(t) & v(t) \end{bmatrix}^T \qquad \xi = \begin{bmatrix} v & \omega \end{bmatrix}^T$$



For single feature point

$$\dot{s} = J_p(u, v, z)\xi$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} -\frac{\lambda}{z} & 0 & \frac{u}{z} & \frac{uv}{\lambda} & -\frac{\lambda^2 + u^2}{\lambda} & v \\ 0 & -\frac{\lambda}{z} & \frac{v}{z} & \frac{\lambda^2 + v^2}{\lambda} & -\frac{uv}{\lambda} & -u \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

N-point Image Feature

$$\dot{s} = J_p(u, v, z)\xi$$

$$s = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix}, \quad z = \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix}$$

$$J_p(s, z) = \begin{bmatrix} J_{p1}(u_1, v_1, z_1) \\ \vdots \\ J_{pn}(u_n, v_n, z_n) \end{bmatrix} = \begin{bmatrix} -\frac{\lambda}{z_1} & 0 & \frac{u_1}{z_1} & \frac{u_1 v_1}{\lambda} & -\frac{\lambda^2 + u_1^2}{\lambda} & v_1 \\ 0 & -\frac{\lambda}{z_1} & \frac{v_1}{z_1} & \frac{\lambda^2 + v_1^2}{\lambda} & -\frac{u_1 v_1}{\lambda} & -u_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -\frac{\lambda}{z_n} & 0 & \frac{u_n}{z_n} & \frac{u_n v_n}{\lambda} & -\frac{\lambda^2 + u_n^2}{\lambda} & v_n \\ 0 & -\frac{\lambda}{z_n} & \frac{v_n}{z_n} & \frac{\lambda^2 + v_n^2}{\lambda} & -\frac{u_n v_n}{\lambda} & -u_n \end{bmatrix}$$

RTSX commands to compute image Jacobian

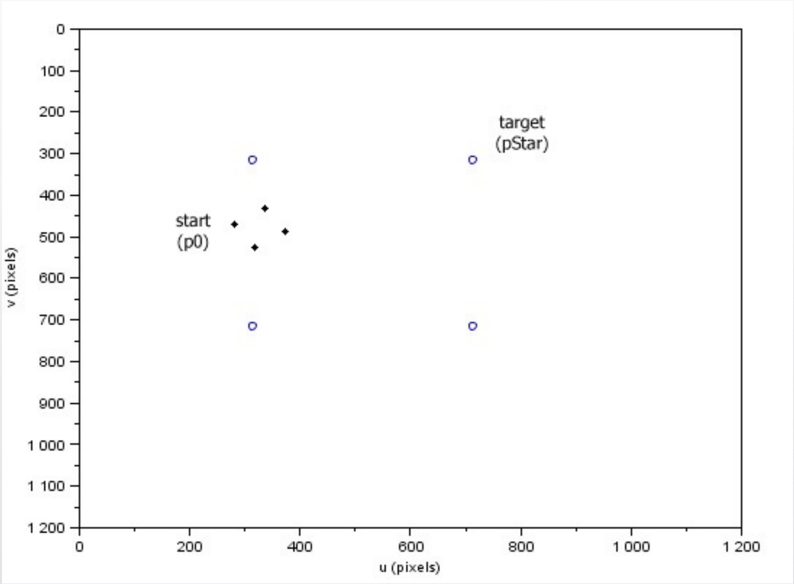
```
cam = CentralCamera('default');

J = visjac_p(cam, [672; 672], 5)

 J  =

 - 160.    0.      32.    32.   - 832.    160.

   0.    - 160.    32.    832.  - 32.   - 160.
```

```
Ps = [150, 250, 400; 200, 350, 600]

 Ps  =

    150.    250.    400.

    200.    350.    600.
```

```
Js = visjac_p(cam, Ps, 1)

 Js  =

 - 800.    0.     - 362.    141.18    - 963.805  - 312.
   0.    - 800.   - 312.    921.68    - 141.18     362.
 - 800.    0.     - 262.    53.055    - 885.805  - 162.
   0.    - 800.   - 162.    832.805   - 53.055     262.
 - 800.    0.     - 112.  - 12.32     - 815.68     88.
   0.    - 800.     88.     809.68      12.32      112.
```

IBVS with proportional control scheme

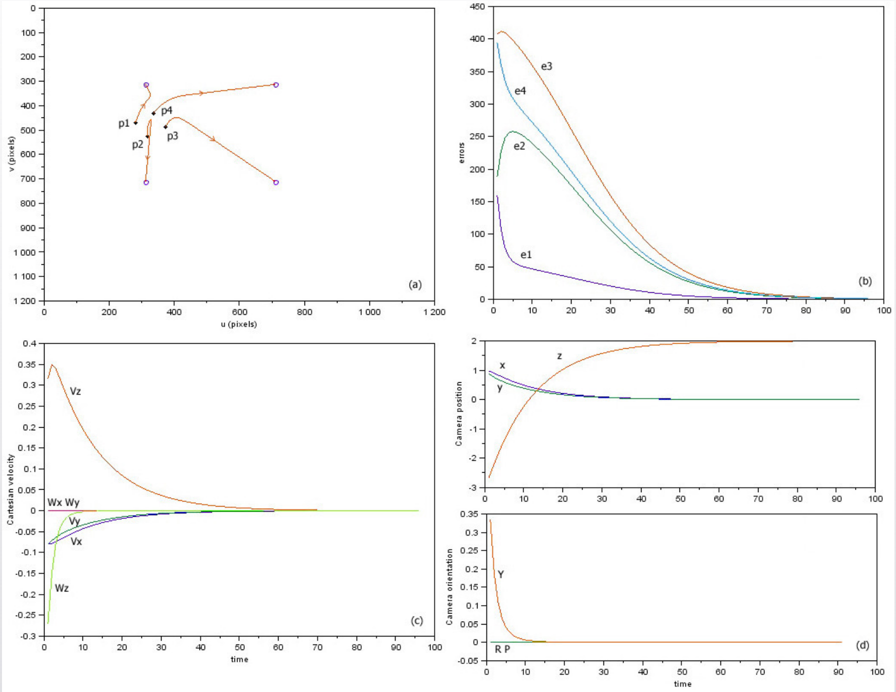
```
cam = CentralCamera('default');

P = mkgrid(2, 0.5, 'T', transl(0,0,3));

pStar = [ 312   312   712   712;
          312   712   712   312];

Tc0 = transl(1,1,-3)*trotz(0.6);

p = CamProject(cam,P,'Tcam',Tc0);

CamPlot(cam,p0,'color','black','figure',1,'size',5);
```




```
CamPlot(cam,pStar,'color','blue','style','o','size',5,'hold');
```

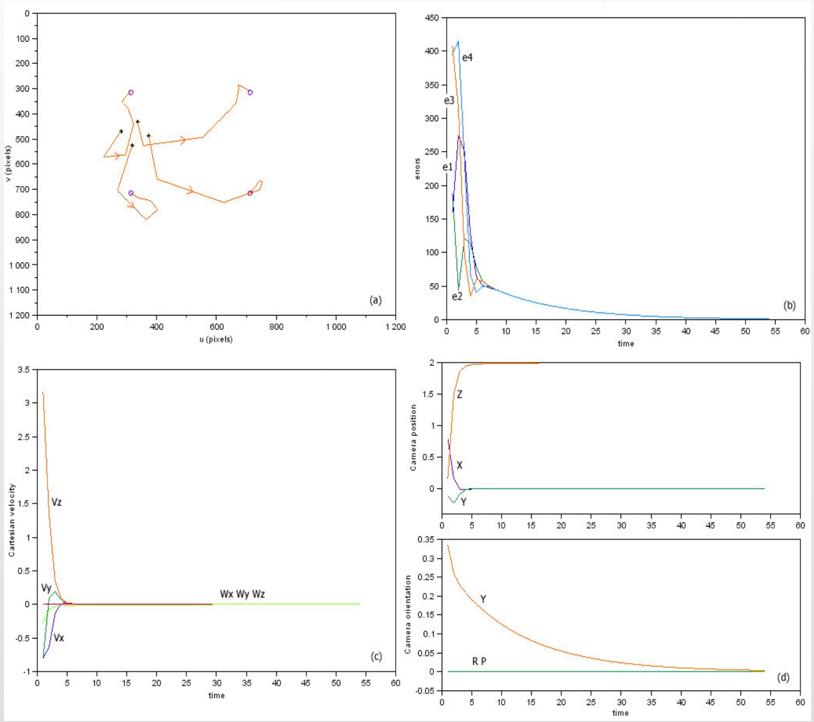
IBVS4 with default depth = 1

```
IBVS4(cam,'T0',Tc0, 'P', P,'pstar',pStar);
```



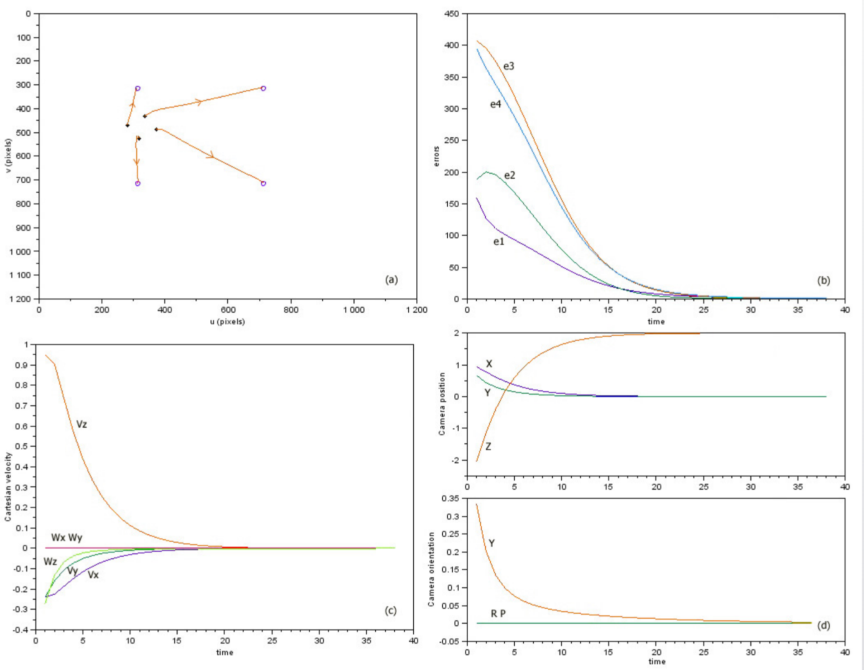
IBVS4 with depth = 10

```
ibvs4(cam,'T0',Tc0, 'P', P,'pstar',pStar,'depth',10);
```



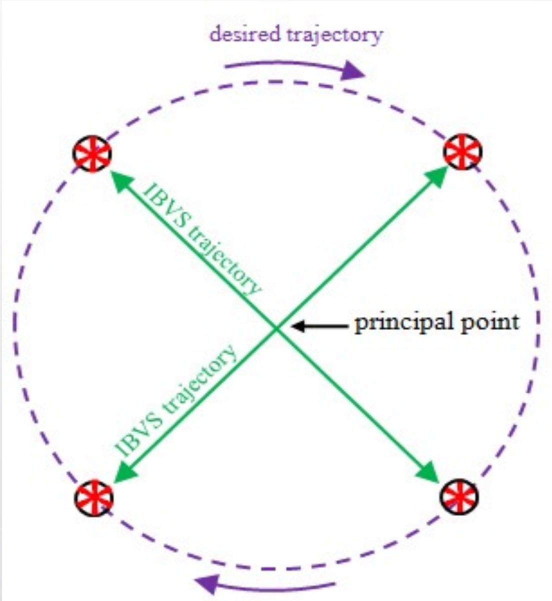
IBVS4 with depth estimation

```
ibvs4(cam,'T0',Tc0,'P',P,'pstar',pStar,'depthest');
```



Performance of vision-based control

```
cam = CentralCamera('default');
P = mkgrid(2, 0.5,'T',transl(0,0,2));
pStar = camproject(cam,P,'Tcam',trotz(0.98*pi));
ibvs4(cam, 'P', P,'pstar',pStar,'depth',2);
```



IBVS4 with Z-axis rotation target