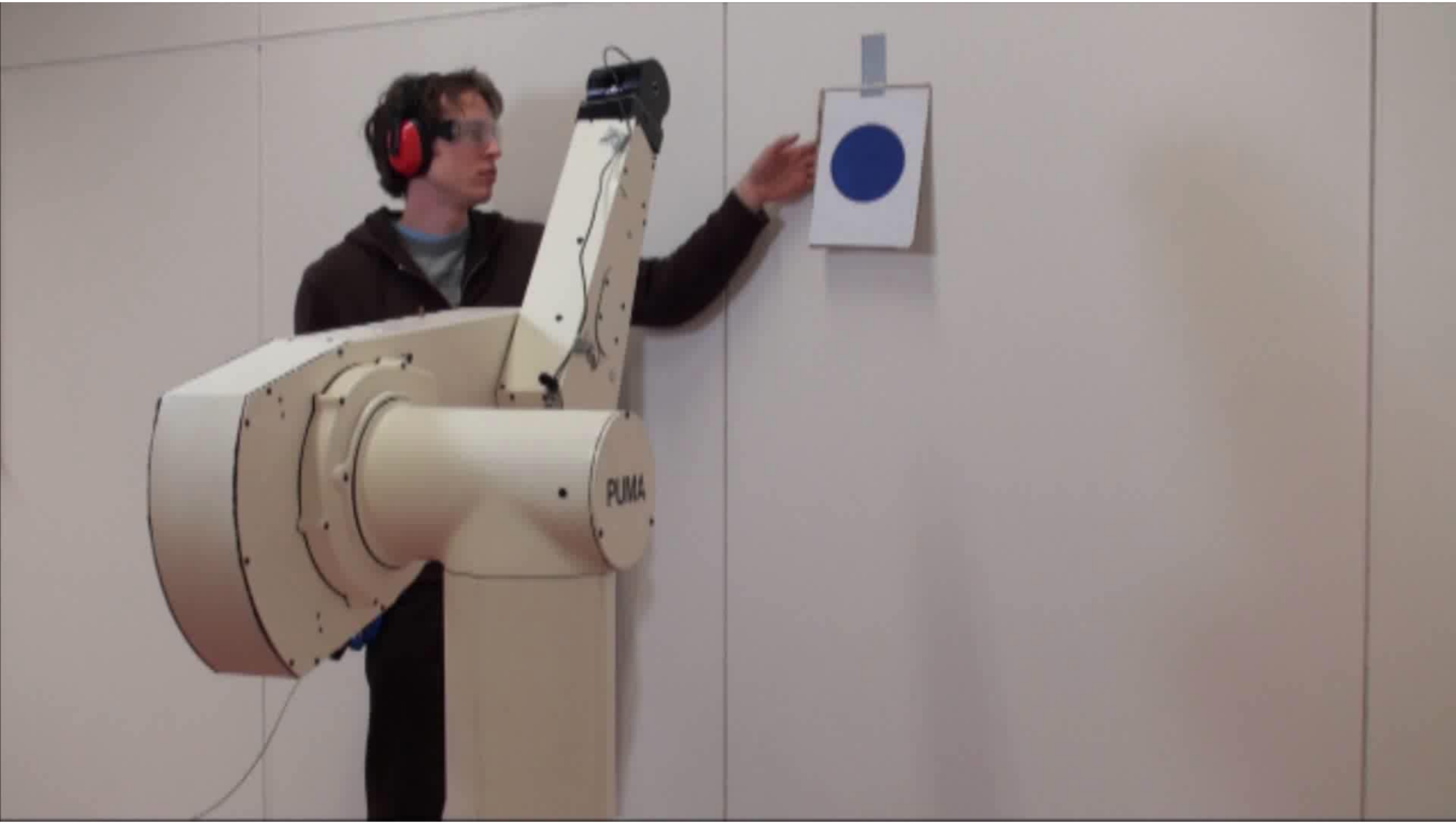
Visual Servoing and Image Jacobian

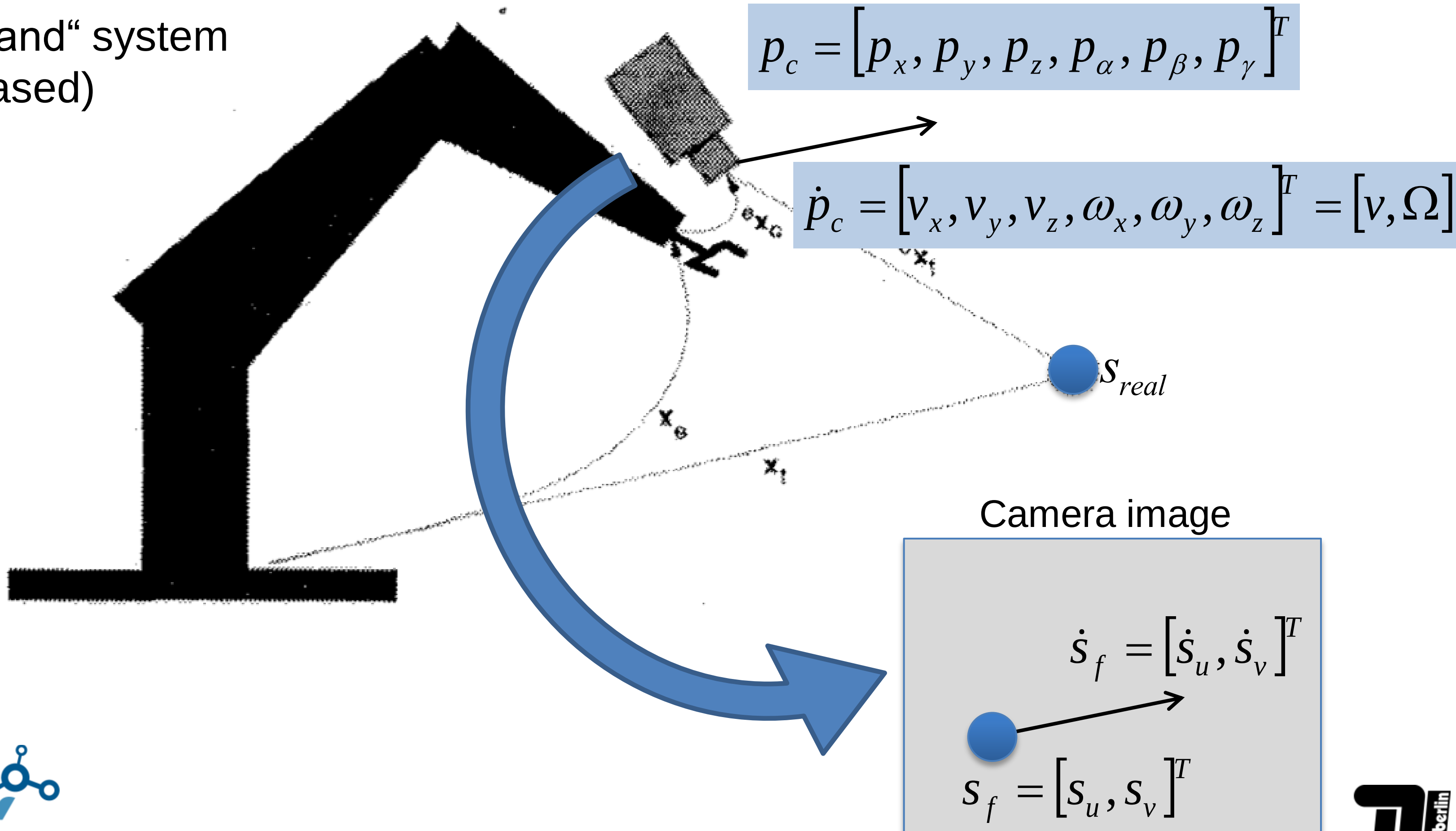
Technische Universität Berlin


PUMA

Motivation

- How does the motion of the end-effector affect the motion of some image feature(s) in the image?

„Eye-in-hand“ system
(image based)



RBO

Image Jacobian

velocity of image features

$$\dot{s}_f = J_I \, \dot{p}_c$$

velocity of camera / EE (linear & angular)

$$J_I(p_c) = \left[\frac{\partial s_f}{\partial p_c} \right] = \begin{bmatrix} \frac{\partial s_{f_1}}{\partial p_{c_1}} & \cdots & \frac{\partial s_{f_1}}{\partial p_{c_m}} \\ \vdots & \ddots & \vdots \\ \frac{\partial s_{f_k}}{\partial p_{c_1}} & \cdots & \frac{\partial s_{f_k}}{\partial p_{c_m}} \end{bmatrix}^{k \times m}$$

dimensionality of task space

dimensionality of feature vector

pose of camera

Example: 2D point feature, 6-DOF task space

We use point features (k = 2) and a task space consisting of translation and rotation about all three axes (m = 6):

$$\begin{bmatrix} \dot{s}_u \\ \dot{s}_v \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

Step 1

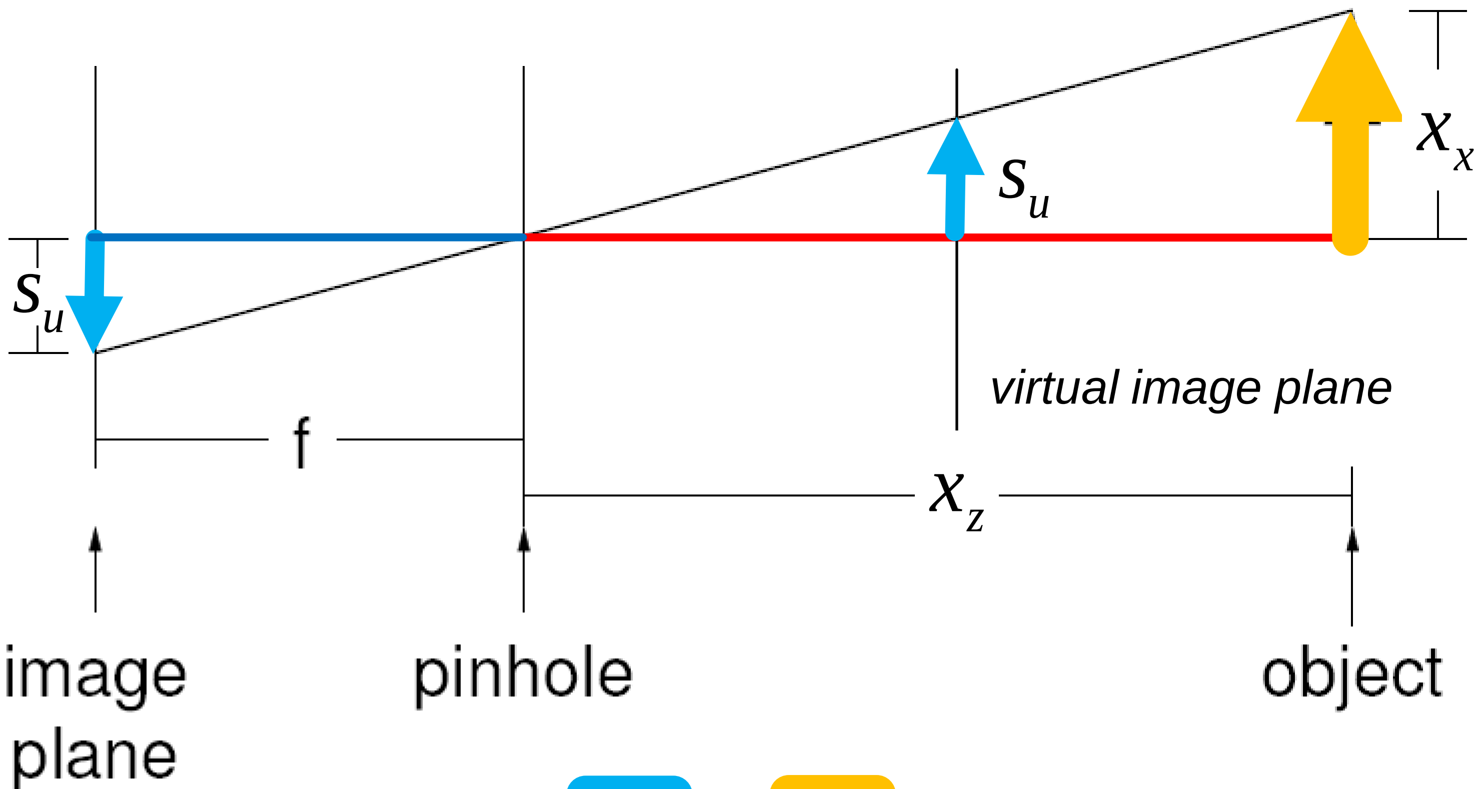
Coordinates of the feature in image space

as a function of

Coordinates of the feature in Cartesian space

$$s = f(x)$$

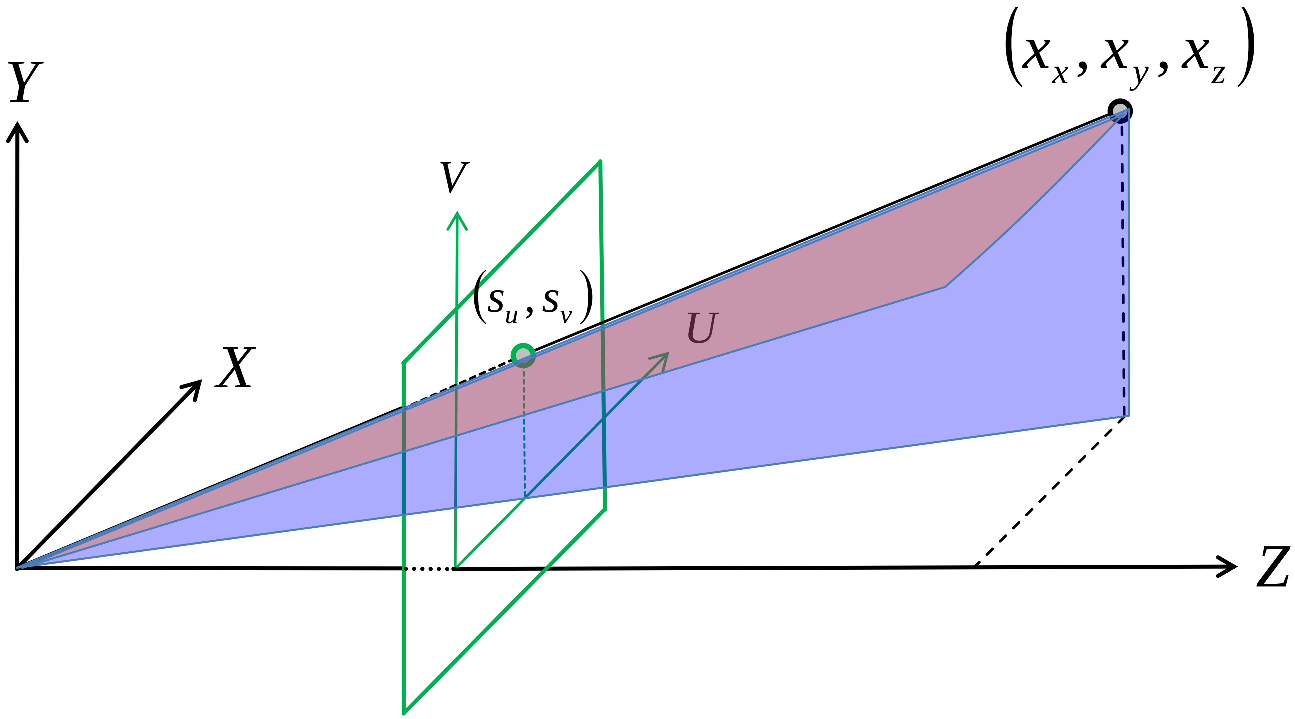
Perspective Projection in 2D



$$\frac{s_u}{f} = \frac{x_x}{x_z} \qquad \Rightarrow s_u = \frac{f}{x_z} x_x$$

focal length

Perspective Projection in 3D



$$s_u = \frac{f}{x_z} x_x$$

$$s_v = \frac{f}{x_z} x_y$$

$$\begin{pmatrix} s_u \\ s_v \end{pmatrix} = \frac{f}{x_z} \begin{pmatrix} x_x \\ x_y \end{pmatrix}$$

Step 2

Velocity of the feature in image space

as a function of

Velocity of the feature in Cartesian space

$$\dot{s} = f(\dot{x})$$

$$s_u = f\,\frac{x_x}{x_z} \qquad \xrightarrow{\frac{d}{dt}} \qquad \dot{s}_u = f\,\frac{x_z\boxed{\dot{x}_x} - x_x\boxed{\dot{x}_z}}{x_z^{\,2}}$$

$$s_v = f\,\frac{x_y}{x_z} \qquad\qquad \dot{s}_v = f\,\frac{x_z\boxed{\dot{x}_y} - x_y\boxed{\dot{x}_z}}{x_z^{\,2}}$$

Step 3

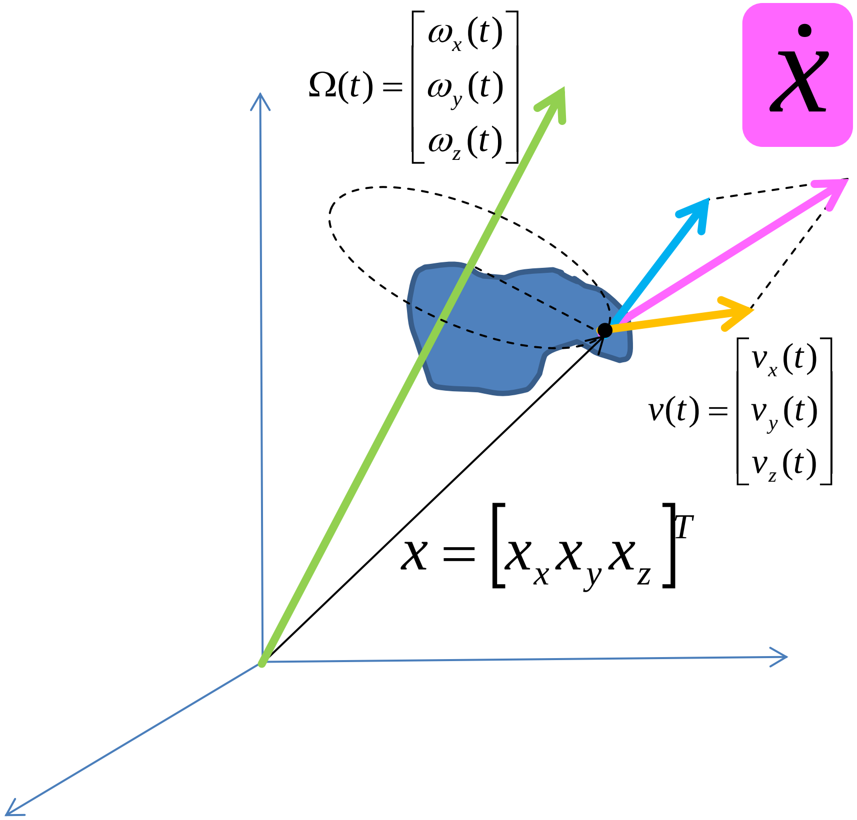
Velocity of the feature in Cartesian space

as a function of

Velocity of the camera in Cartesian space

$$\dot{x} = f(\dot{p}_c) = f(v, \Omega)$$

Reminder: Linear and angular velocity



$$\dot{x} = \Omega \times x + v = \begin{bmatrix} \dot{x}_x \\ \dot{x}_y \\ \dot{x}_z \end{bmatrix} = \begin{bmatrix} x_z\omega_y - x_y\omega_z + v_x \\ x_x\omega_z - x_z\omega_x + v_y \\ x_y\omega_x - x_x\omega_y + v_z \end{bmatrix}$$

$$\dot{x} = -\Omega \times x - v$$

Negative velocity as we move the frame, not the feature!!!!

$$\dot{x} = \begin{bmatrix} \dot{x}_x \\ \dot{x}_y \\ \dot{x}_z \end{bmatrix} = \begin{bmatrix} -x_z\omega_y + x_y\omega_z - v_x \\ -x_x\omega_z + x_z\omega_x - v_y \\ -x_y\omega_x + x_x\omega_y - v_z \end{bmatrix} = \begin{bmatrix} -x_z\omega_y + \frac{s_v x_z}{f}\omega_z - v_x \\ -\frac{s_u x_z}{f}\omega_z + x_z\omega_x - v_y \\ -\frac{x_z}{f}\left(s_v\omega_x - s_u\omega_y\right) - v_z \end{bmatrix}$$

$$\begin{pmatrix} x_x \\ x_y \end{pmatrix} = \frac{x_z}{f}\begin{pmatrix} s_u \\ s_v \end{pmatrix}$$

Step 4 -> All together! -> Image Jacobian!

Velocity of the feature in image space

as a function of

Velocity of the camera in Cartesian space

$$\dot{s} = f(\dot{p}_c) = f(v, \Omega) = \boxed{J_I \dot{p}_c}$$

Step3

$$\dot{x}_x = -x_z\omega_y + \frac{s_v x_z}{f}\omega_z - v_x$$

$$\dot{x}_y = -\frac{s_u x_z}{f}\omega_z + x_z\omega_x - v_y$$

$$\dot{x}_z = -\frac{x_z}{f}\left(s_v\omega_x - s_u\omega_y\right) - v_z$$

Step1 $$s_u = f\,\frac{x_x}{x_z}$$

Step2 $$\dot{s}_u = f\,\frac{x_z\dot{x}_x - x_x\dot{x}_z}{x_z^{\,2}} =$$

$$= \frac{f}{x_z^{\,2}}\left[x_z\left(-x_z\omega_y + \frac{s_v x_z}{f}\omega_z - v_x\right) - \frac{s_u x_z}{f}\left(-\frac{x_z}{f}\left(s_v\omega_x - s_u\omega_y\right) - v_z\right)\right]$$

$$\dot{s}_u = -\frac{f}{x_z}v_x + \frac{s_u}{x_z}v_z + \frac{s_u s_v}{f}\omega_x - \frac{f^2 + s_u^{\,2}}{f}\omega_y + s_v\omega_z$$

$$\dot{s}_v = -\frac{f}{x_z}v_y + \frac{s_v}{x_z}v_z + \frac{f^2 + s_v^{\,2}}{f}\omega_x - \frac{s_u s_v}{f}\omega_y - s_u\omega_z$$

Example: 2D point feature, 6-DOF task space

$$\begin{bmatrix} \dot{s}_u \\ \dot{s}_v \end{bmatrix} = \begin{bmatrix} ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

Image Jacobian for a Point Feature

$$\begin{bmatrix} \dot{s}_u \\ \dot{s}_v \end{bmatrix} = \begin{bmatrix} -\frac{f}{x_z} & 0 & \frac{s_u}{x_z} & \frac{s_u s_v}{f} & -\frac{f^2 + s_u^2}{f} & s_v \\ 0 & -\frac{f}{x_z} & \frac{s_v}{x_z} & \frac{f^2 + s_v^2}{f} & -\frac{s_u s_v}{f} & -s_u \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

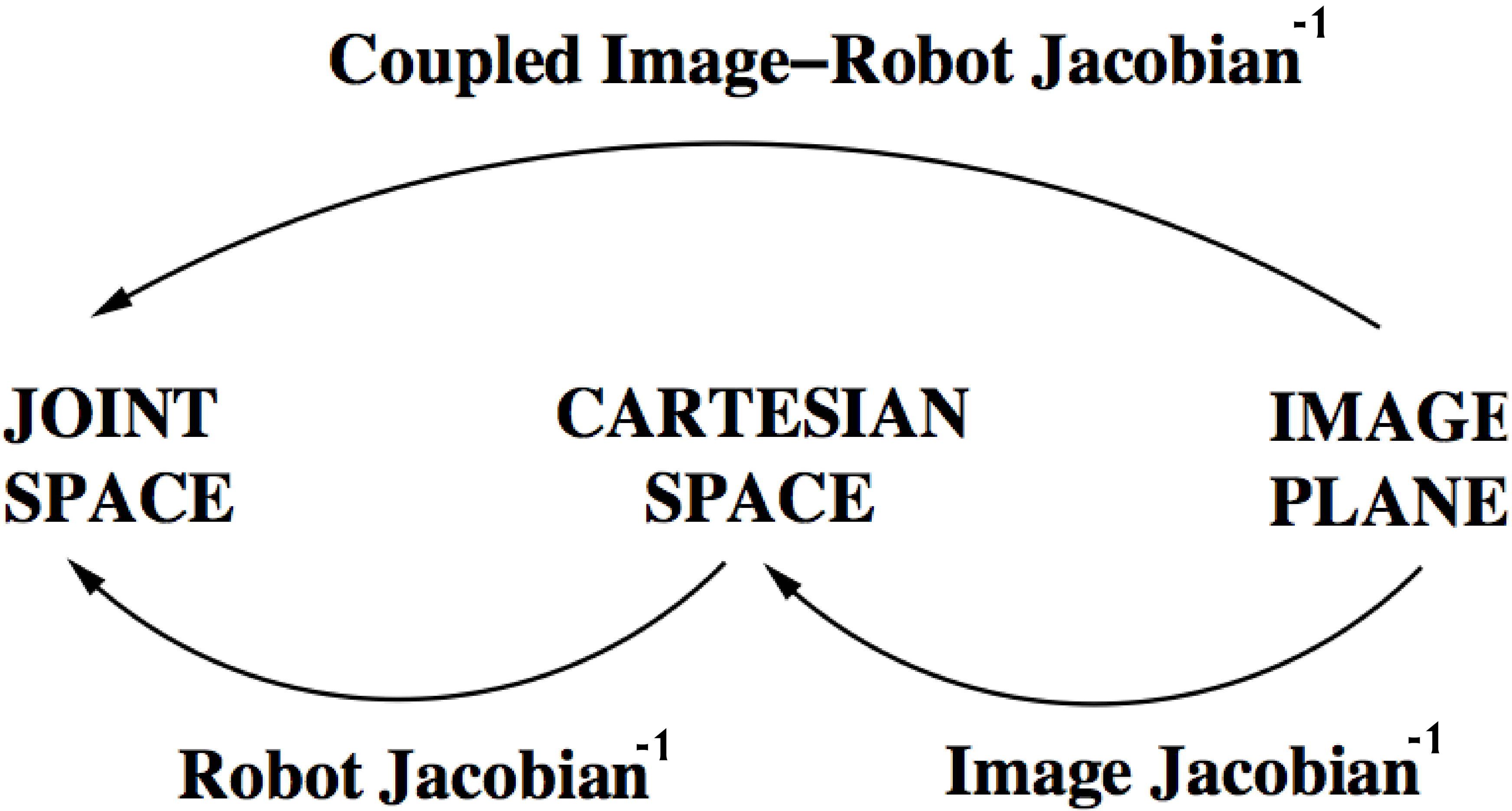
Using J_I to compute the camera/end-effector velocity

$$\dot{p}_c = J_I^{-1} \dot{s}_f$$

- Inverse only exists if J_I is square (k = m) and nonsingular
- If k ≠ m:
  system is either overconstrained (k > m) or underconstrained (k < m)

  ➔ Choose more features (e.g. stack J_I's) or use Pseudoinverse (if J_I is full rank)

Robot Control: Two Jacobians!



Singularities

- How to check if the Jacobian is singular?
  - Determinant, Rank, Eigenvalues,...
- When does the Image Jacobian become singular?