

CSE276A - Visual Servoing

Henrik I Christensen

HW1: Deadline tonight

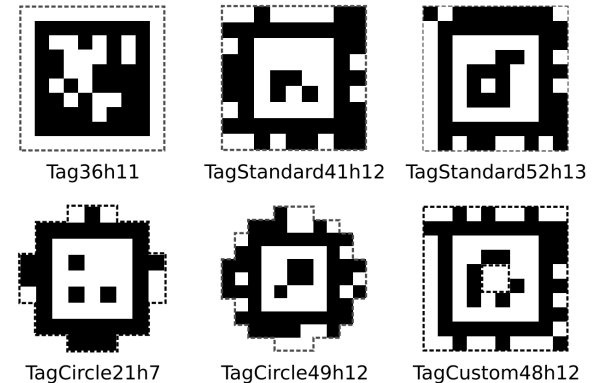
- Too much friction - consider taping wheels or different surface
- Report
 - What is the challenge
 - Algorithm
 - What is missing? Determined how / calibration
 - Resulting solution
 - Test results
 - YouTube Link to video
- See last lecture about possible ways to send commands

HW2 - Closing the loop with vision

- Due: 29 October @ midnight
- Two options:
 - Use AprilTags as landmarks in the environment
 - Use natural landmarks for extra credit (it is hard!)
 - Tutorial on how to use DL for detection of objects say Yolo
 - <https://pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/>
 - 20% extra credit
- Use detected landmarks to estimate your own position
 - Correct for drift in the control of the vehicle

(c) Henrik I Christensen

AprilTags

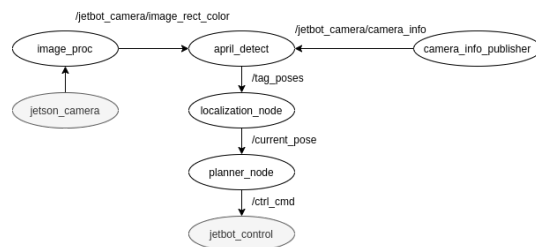


<https://github.com/AprilRobotics/apriltag>

(c) Henrik I Christensen

Practical stuff

- You have to calibrate your camera !
- OpenCV see later
- ROS camera calibration
- Example process with ROS modules



(c) Henrik I Christensen

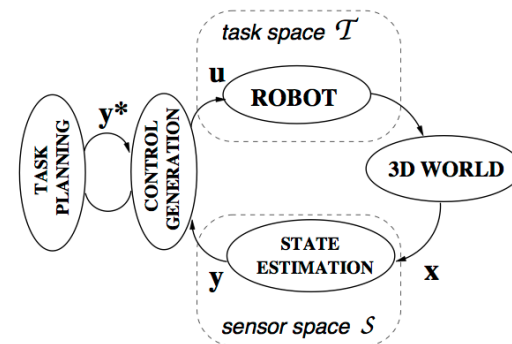
Visual Servoing

Henrik I Christensen
hichristensen@ucsd.edu

Literature

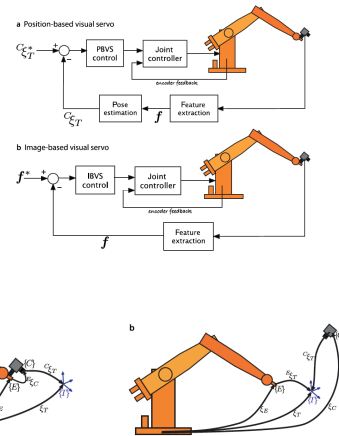
- Peter Corke, Robotics - Vision and Control, Springer Verlag, 2nd Edition, 2023 - Chapter 15

Major Robot Processes

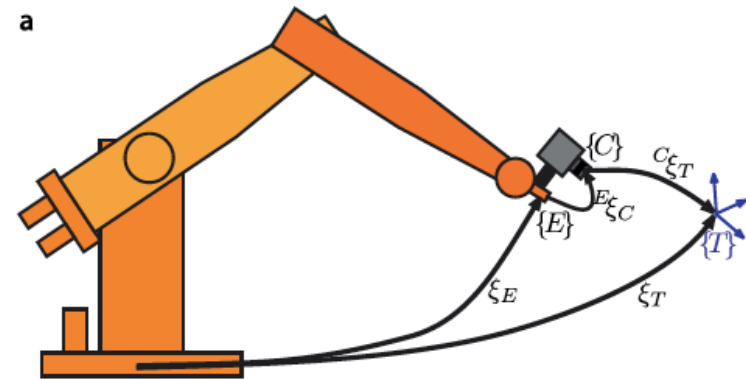


Types of visual serving

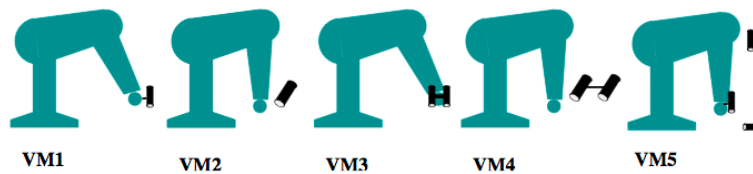
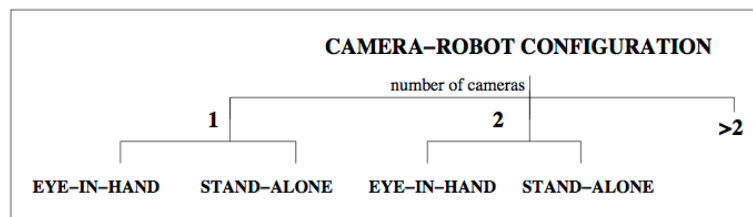
- Control Strategy
 - Position based servoing
 - Image based servoing
- Camera Placement
 - Eye-in-Hand
 - Stand-alone



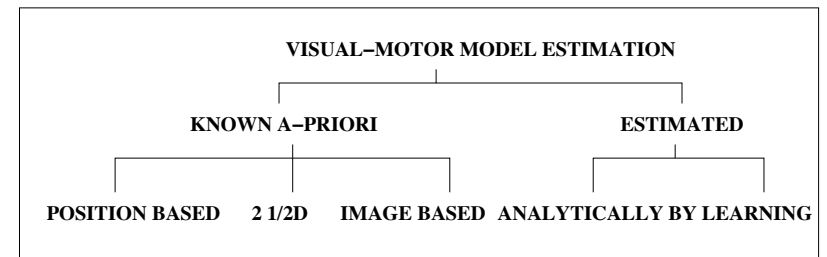
Process



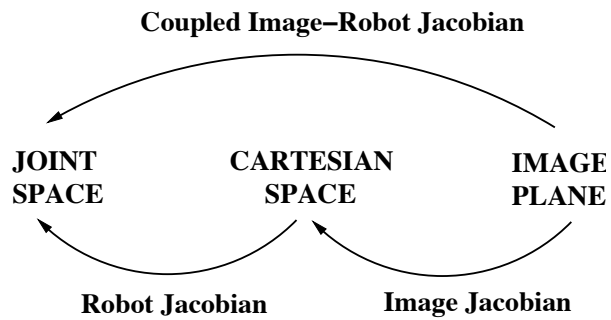
Categorization



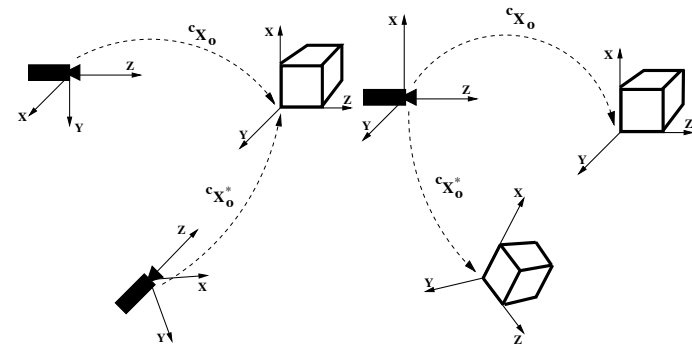
Visual Model Estimation



Utilization of kinematic models



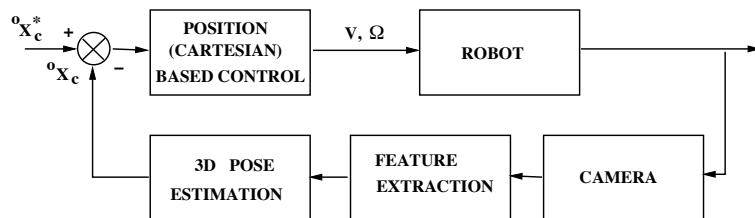
Example of position based visual servoing



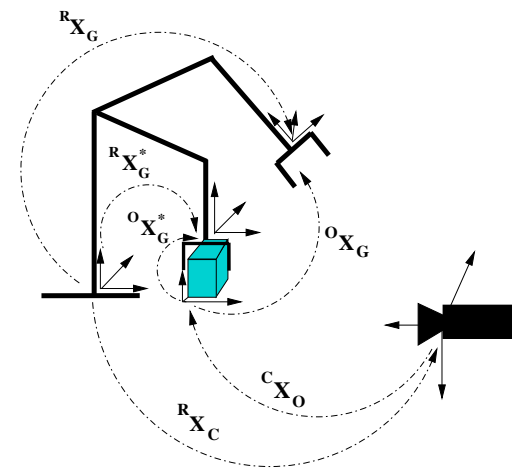
Moving Camera

Moving Object

Position based visual serving



Relevant reference frames



Control

- Error Metric

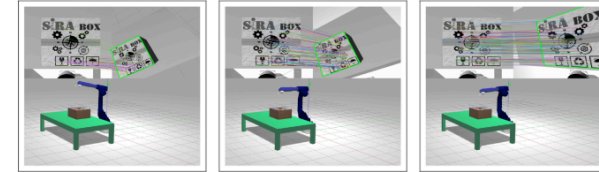
$$\Delta^R \mathbf{t}_G = {}^R \mathbf{t}_G - {}^R \mathbf{t}_G^*$$

$$\Delta^R \theta_G = {}^R \theta_G - {}^R \theta_G^*$$

- Relevant Coordinate frames

$${}^R \mathbf{X}_G^* = {}^R \mathbf{X}_C {}^C \mathbf{X}_O {}^O \mathbf{X}_G^*$$

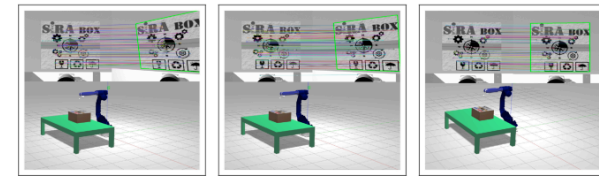
PBS Cartoon



(a) Initial position of the robot. Position a in Fig. 13.

(b) Position b in Fig. 13.

(c) Position c in Fig. 13.



(d) Position d in Fig. 13.

(e) Position e in Fig. 13.

(f) Final position of the robot. Position f in Fig. 13.

Expanding the error metrics

- Expanding the transformation

$${}^R \mathbf{t}_G^* = {}^R \mathbf{R}_C {}^C \hat{\mathbf{R}}_O {}^O \mathbf{t}_G^* + {}^R \mathbf{R}_C {}^C \hat{\mathbf{t}}_O + {}^R \mathbf{t}_C$$

$${}^R \Omega_G^* = {}^R \Omega_C + {}^R \mathbf{R}_C {}^C \hat{\Omega}_O + {}^R \mathbf{R}_C {}^C \hat{\mathbf{R}}_O {}^O \Omega_G^*$$

- Integrating

$${}^R \theta_G^* \approx {}^R \theta_C + {}^R \mathbf{R}_C {}^C \hat{\theta}_O + {}^R \mathbf{R}_C {}^C \hat{\mathbf{R}}_O {}^O \theta_G^*$$

- Substituting from above

$$\Delta^R \mathbf{t}_G = {}^R \mathbf{t}_G - {}^R \mathbf{t}_C - {}^R \mathbf{R}_C {}^C \hat{\mathbf{t}}_O - {}^R \mathbf{R}_C {}^C \hat{\mathbf{R}}_O {}^O \mathbf{t}_G^*$$

$$\Delta^R \theta_G \approx {}^R \theta_G - {}^R \theta_C - {}^R \mathbf{R}_C {}^C \hat{\theta}_O - {}^R \mathbf{R}_C {}^C \hat{\mathbf{R}}_O {}^O \theta_G^*$$

Strategy

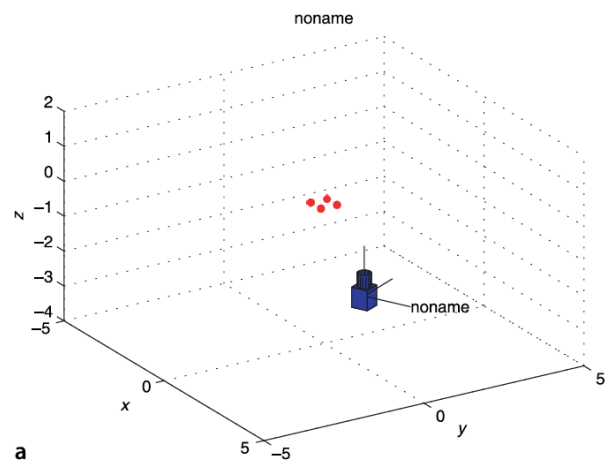
- Compute

$$\mathbf{e} = \begin{bmatrix} \Delta^R \mathbf{t}_G \\ \Delta^R \theta_G \end{bmatrix}$$

- The change in robot coordinates (use, PID, ...)

$$\dot{\mathbf{q}} \approx \mathbf{K} \mathbf{e}$$

Simulated camera



Example PBS simulation

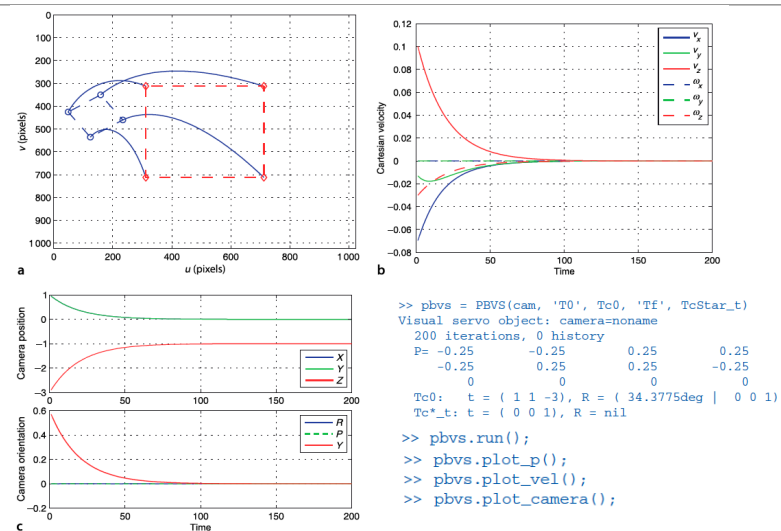
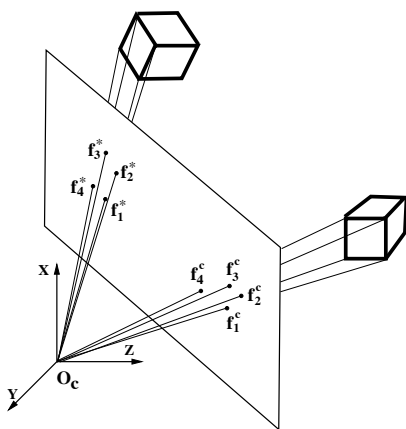
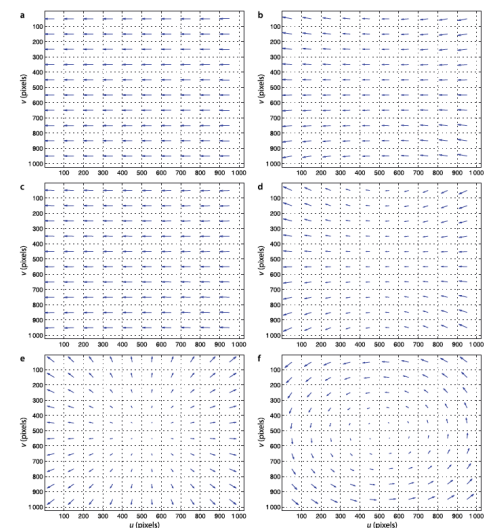


Image Based Visual Servoing (IBVS)



Canonical image motion patterns



The Image Jacobian

- How does features move when we move the robot?

$$\dot{\mathbf{f}} = \mathbf{J} \dot{\mathbf{q}}$$

- Image Jacobian

$$\mathbf{J}(\mathbf{q}) = \left[\frac{\delta \mathbf{f}}{\delta \mathbf{q}} \right] = \begin{bmatrix} \frac{\delta f_1(\mathbf{q})}{\delta q_1} & \cdots & \frac{\delta f_1(\mathbf{q})}{\delta q_m} \\ \vdots & \ddots & \vdots \\ \frac{\delta f_k(\mathbf{q})}{\delta q_1} & \cdots & \frac{\delta f_k(\mathbf{q})}{\delta q_m} \end{bmatrix}$$

Going back to basics

$$x = \frac{X}{Z}, y = \frac{Y}{Z}$$

$$\dot{x} = \frac{\dot{X}Z - X\dot{Z}}{Z^2}, \dot{y} = \frac{\dot{Y}Z - Y\dot{Z}}{Z^2}$$

$$\dot{\mathbf{P}} = -\boldsymbol{\omega} \times \mathbf{P} - \mathbf{v}$$

$$\dot{X} = Y\omega_z - Z\omega_y - v_x$$

$$\dot{Y} = Z\omega_x - X\omega_z - v_y$$

$$\dot{Z} = X\omega_y - Y\omega_x - v_z$$

Going back to basics

- With a unit focal length we would have

$${}^c\dot{\mathbf{P}} = {}^c\boldsymbol{\Omega} \times {}^c\mathbf{P} + {}^c\mathbf{V}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} k_u & 0 \\ 0 & k_v \end{bmatrix} \begin{bmatrix} \frac{1}{Z} & 0 & -\frac{x}{Z} & -xy & 1+x^2 & -y \\ 0 & \frac{1}{Z} & -\frac{y}{Z} & -1-y^2 & xy & x \end{bmatrix} \begin{bmatrix} V_X \\ V_Y \\ V_Z \\ \omega_X \\ \omega_Y \\ \omega_Z \end{bmatrix}$$

- We can define an error metric

$$\mathbf{e}(\mathbf{f}) = \mathbf{f}^c - \mathbf{f}^*$$

$$\mathbf{u} = \dot{\mathbf{q}} = \mathbf{KJ}^\dagger \mathbf{e}(\mathbf{f})$$

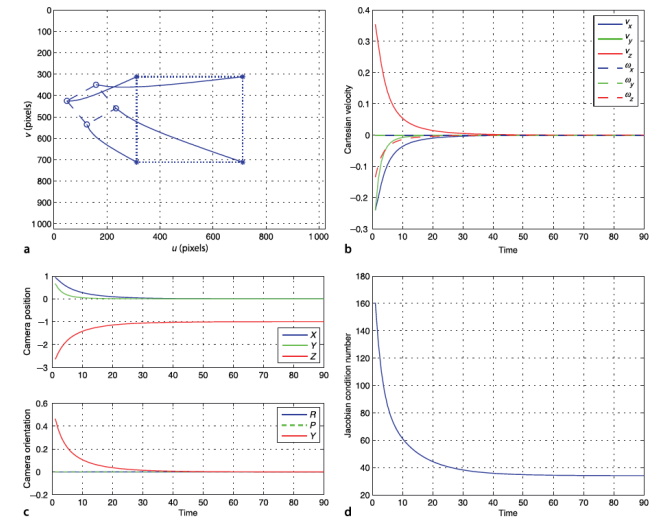
For K points we can stack

$$\begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \vdots \\ \dot{u}_k \\ \dot{v}_k \end{bmatrix} = \mathbf{A} \begin{bmatrix} \frac{1}{Z_1} & 0 & -\frac{x_1}{Z_1} & -x_1y_1 & 1+x_1^2 & -y_1 \\ 0 & \frac{1}{Z_1} & -\frac{y_1}{Z_1} & -1-y_1^2 & x_1y_1 & x_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{Z_k} & 0 & -\frac{x_k}{Z_k} & -x_ky_k & 1+x_k^2 & -y_k \\ 0 & \frac{1}{Z_k} & -\frac{y_k}{Z_k} & -1-y_k^2 & x_ky_k & x_k \end{bmatrix} \begin{bmatrix} V_X \\ V_Y \\ V_Z \\ \omega_X \\ \omega_Y \\ \omega_Z \end{bmatrix}$$

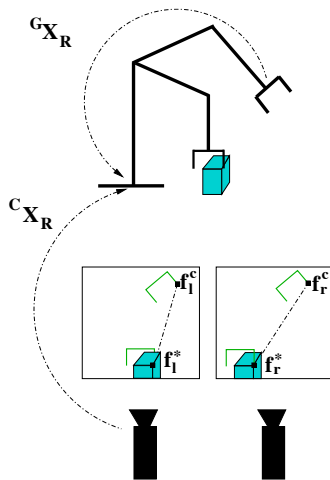
Visual impression of image jacobian



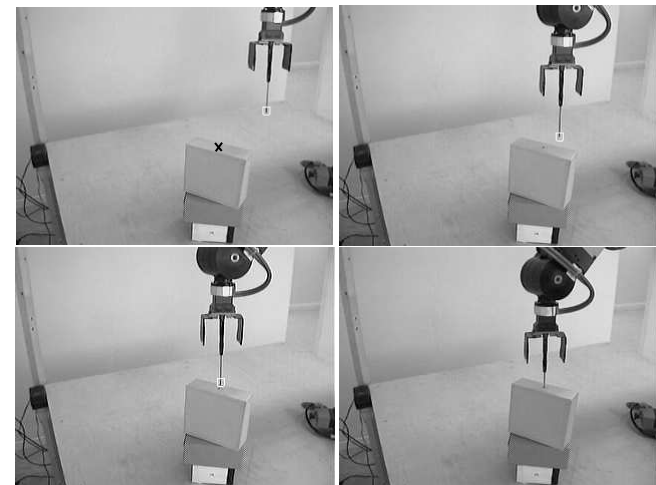
Simulated example



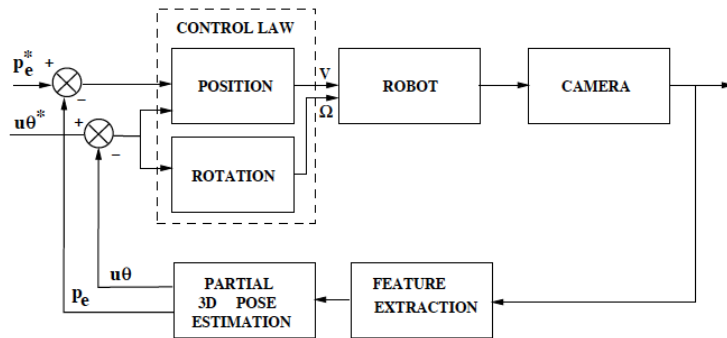
Sketch of an IBVS example



IBVS Cartoon



2.5D image based servoing



What if we could estimate Z online?

$$\begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} -\frac{f}{\rho_u Z} & 0 & \frac{\bar{u}}{Z} \\ 0 & -\frac{f}{\rho_v Z} & \frac{\bar{v}}{Z} \end{pmatrix} \begin{pmatrix} \frac{\rho_u \bar{u} \bar{v}}{f} & -\frac{f^2 + \rho_u^2 \bar{u}^2}{\rho_u f} & \bar{v} \\ \frac{f^2 + \rho_v^2 \bar{v}^2}{\rho_v f} & -\frac{\rho_v \bar{u} \bar{v}}{f} & -\bar{u} \end{pmatrix} \begin{pmatrix} v \\ \omega \end{pmatrix}$$

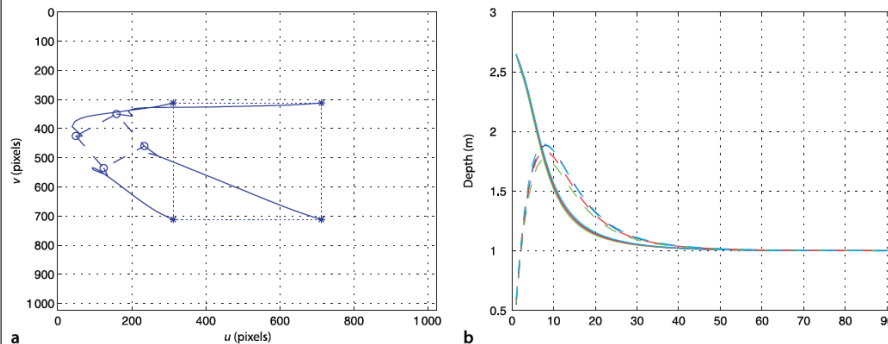
$$= \left(\frac{1}{Z} J_t \mid J_\omega \right) \begin{pmatrix} v \\ \omega \end{pmatrix}$$

$$= \frac{1}{Z} J_t v + J_\omega \omega$$

- We can rearrange

$$(J_t v) \frac{1}{Z} = \begin{pmatrix} \dot{u} \\ \dot{v} \end{pmatrix} - J_\omega \omega \quad \Rightarrow \quad \mathbf{A} \boldsymbol{\theta} = \mathbf{b}$$

Example from the book



Summary / Take Home

- Image vs. position based servoing
- Derivation of motion impact on image coordinates
- Use of basic motion models to control the robot
- Great support for PBS and IBVS in the RVT toolkit