Using 2D projective geometry to control of robot motion from video "visual servoing"

M. Jagersand

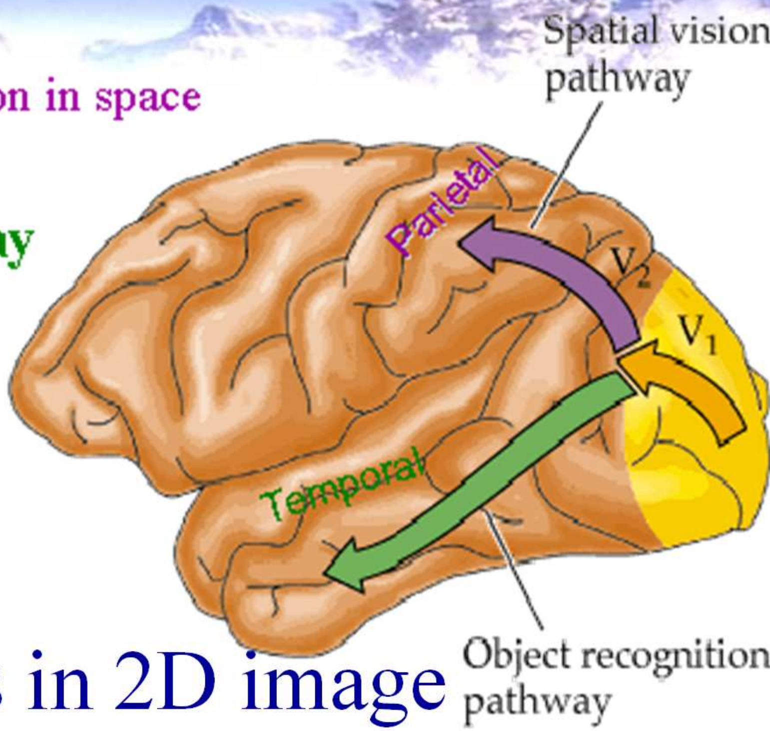
What is vision?

Dorsal (magno) Pathway

- to parietal lobe
- spatial vision – localization in space
- "WHERE"

Ventral (parvo) Pathway

- to temporal lobe
- object recognition
- "WHAT"

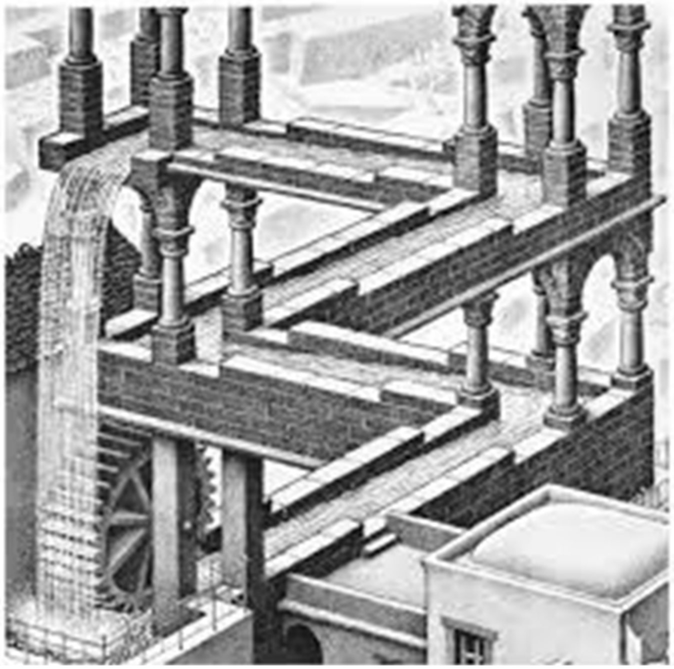


Two main categories

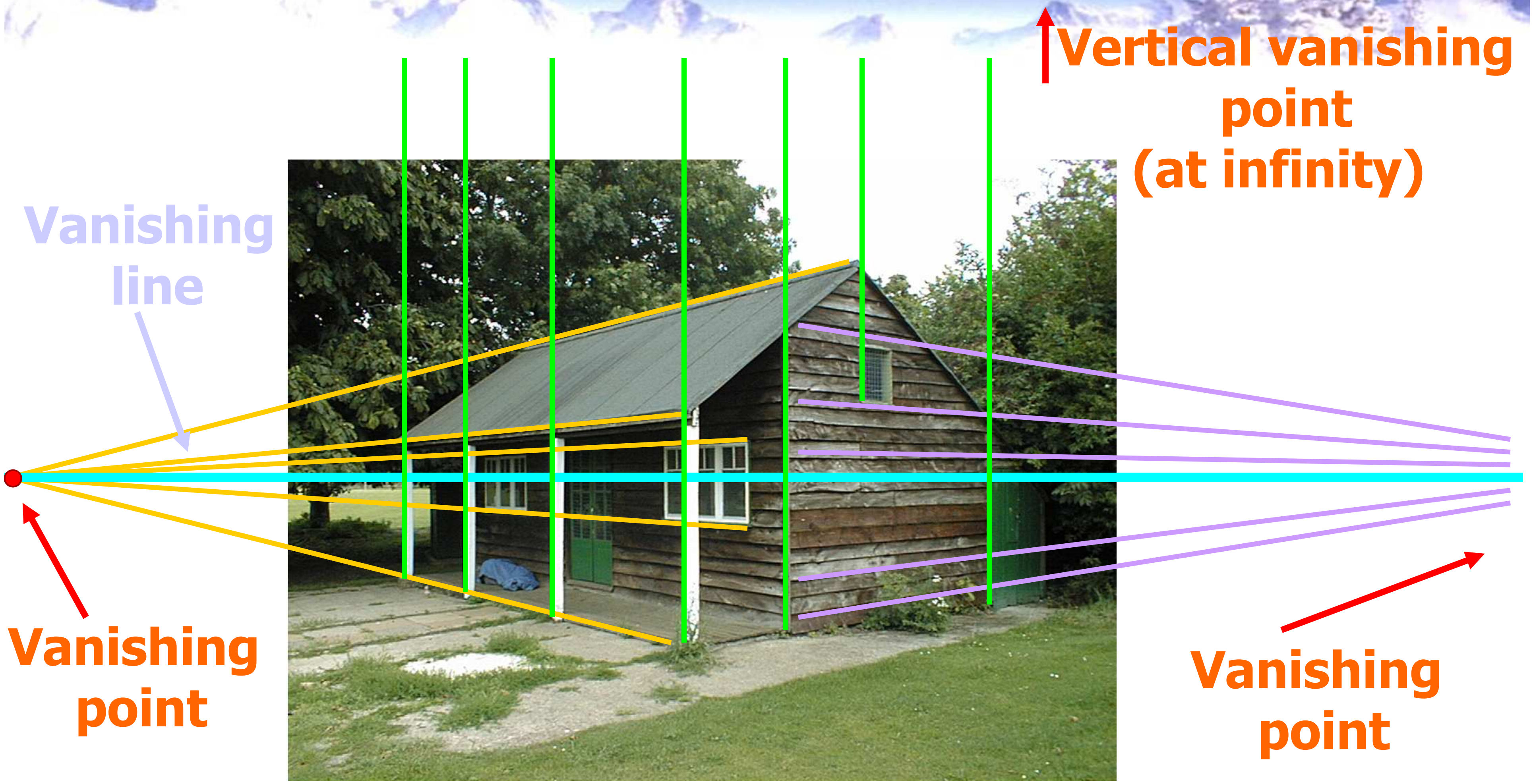
- Object recognition: Use cues in 2D image
- Spatial vision: Understand and invert the 3D to 2D imaging process: Two subcategories:
  - Full 3D scene reconstruction
  - Use of partial 3D information, implicitly or explicitly

Uses of partial information from 2D-3D camera geometry

- Single view measurements
- Visual constraints: verify alignments, detect impossible configurations (Esher paintings)
- Visual servoing
  - Lukas Kanade –like
  - Deep Learning
- Visual tracking
  - Features instead of images
- … many more



Geometric Cues

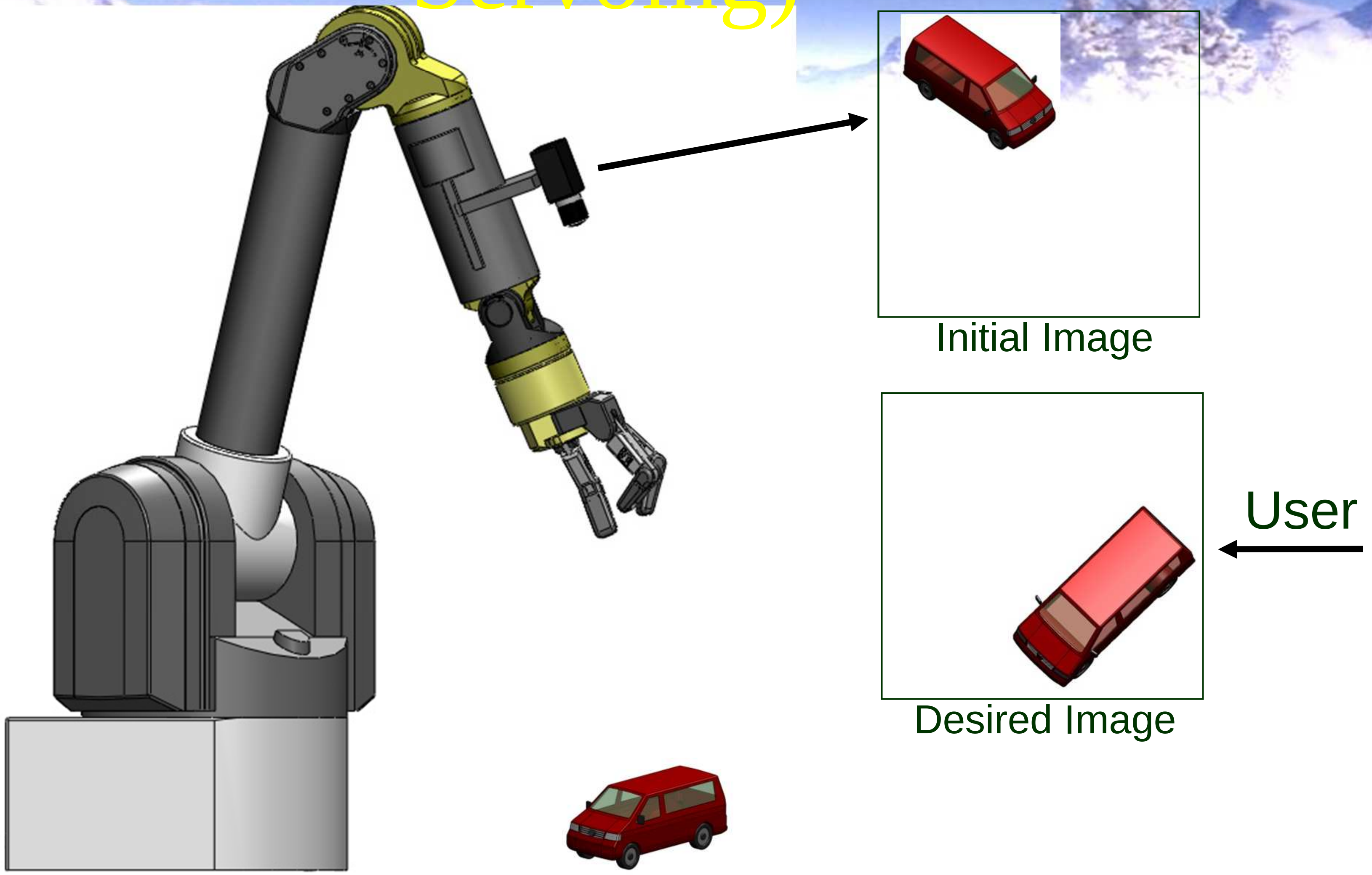


Single View Metrology

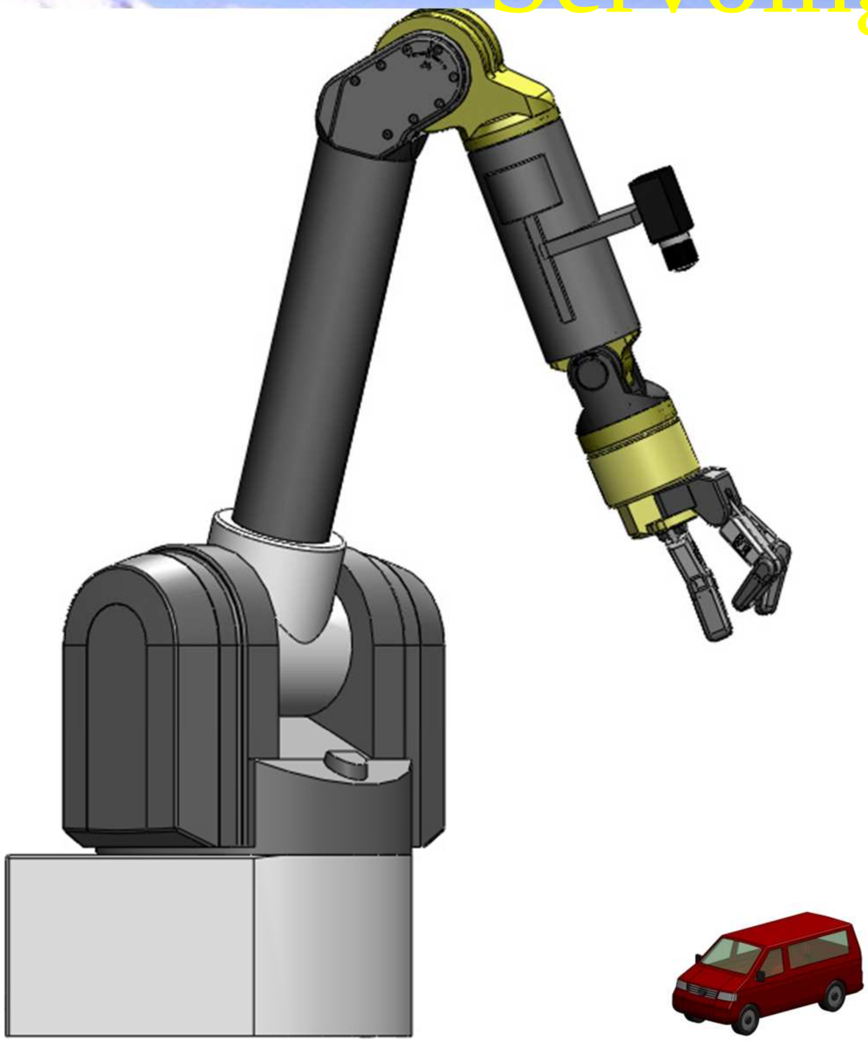
Estimating Height

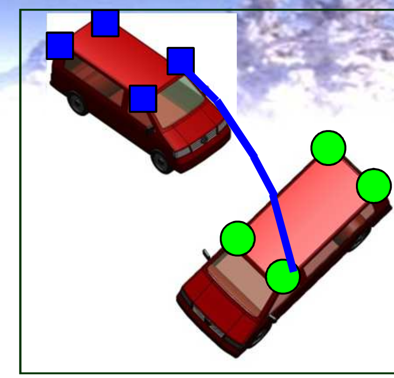
- The distance $|| t_r - b_r ||$ is known
- Used to estimate the height of the man in the scene

Vision-Based Control (Visual Servoing)



Vision-Based Control (Visual Servoing)

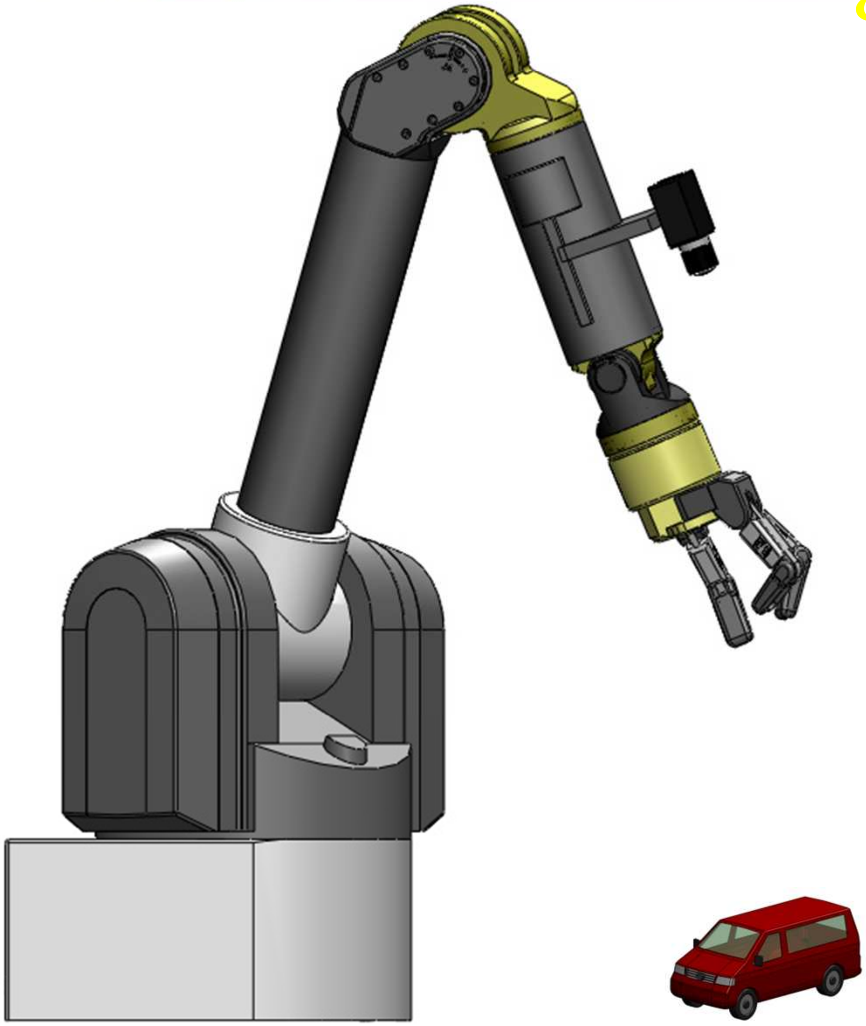


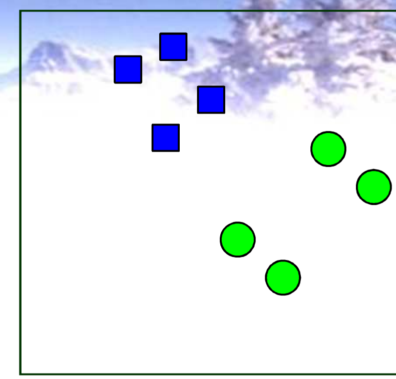


■ : Current Image Features

● : Desired Image Features

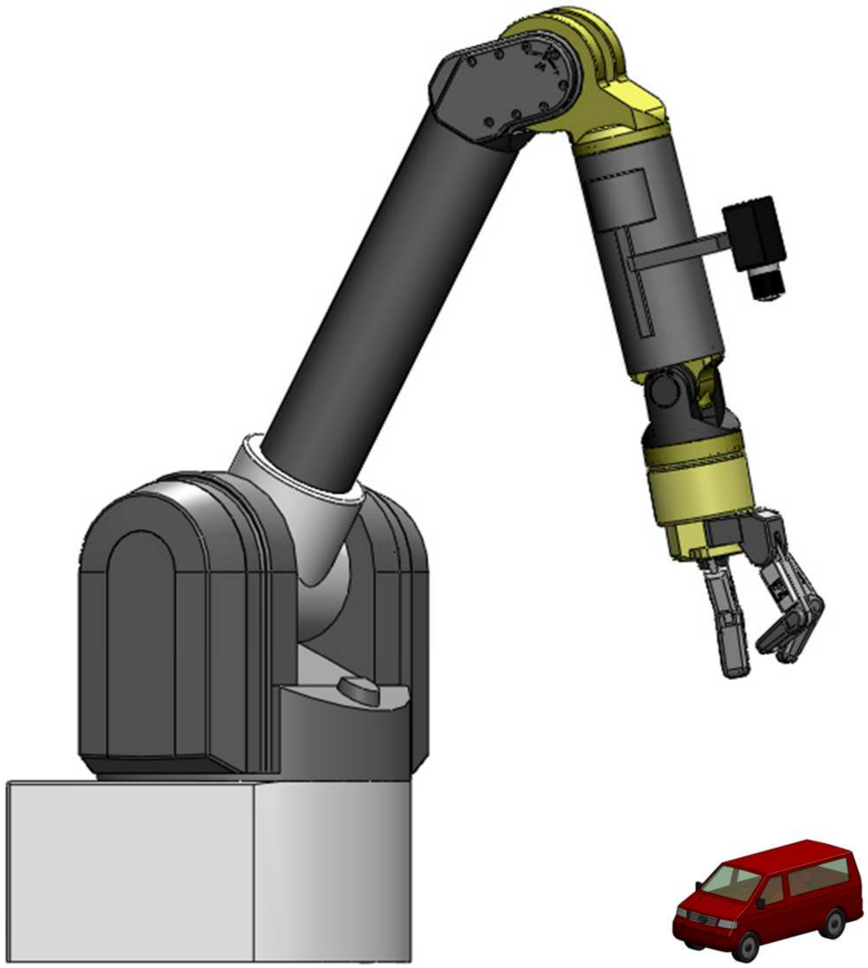
Vision-Based Control (Visual Servoing)

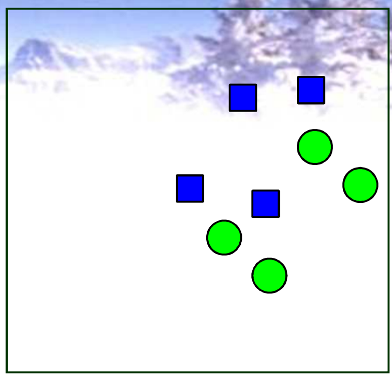




■ : Current Image Features
● : Desired Image Features

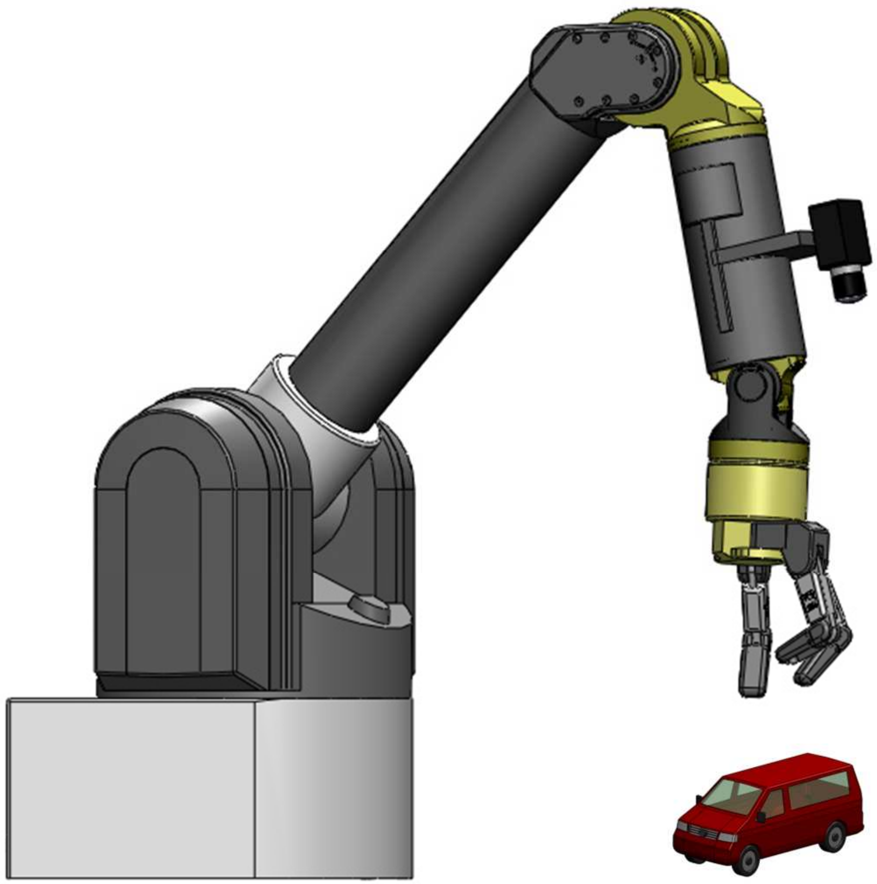
Vision-Based Control (Visual Servoing)

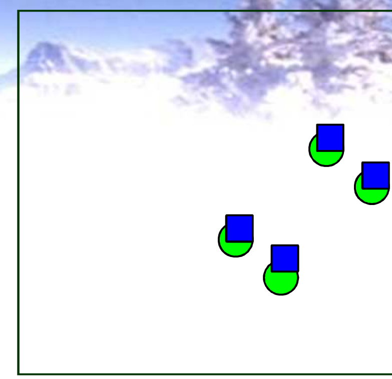




■ : Current Image Features
● : Desired Image Features

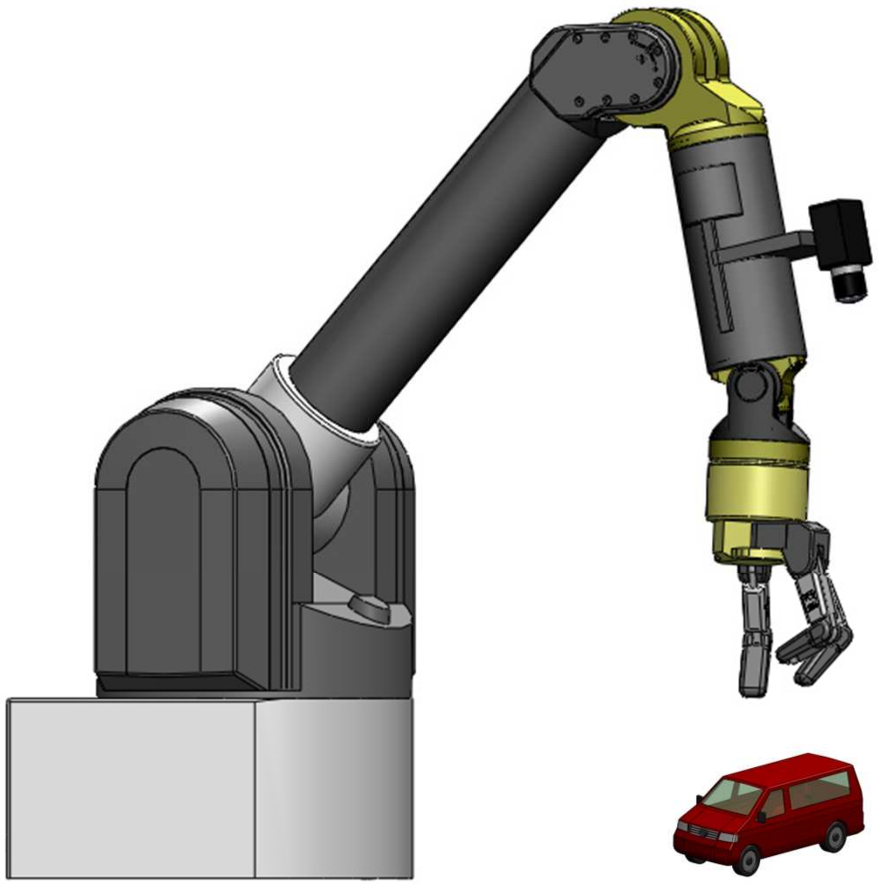
Vision-Based Control (Visual Servoing)

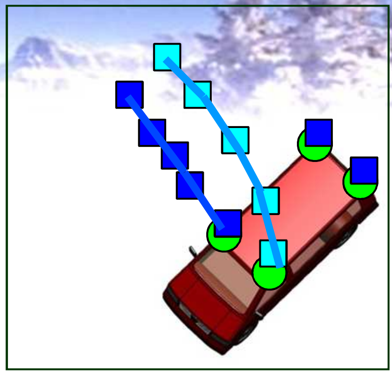




: Current Image Features
: Desired Image Features

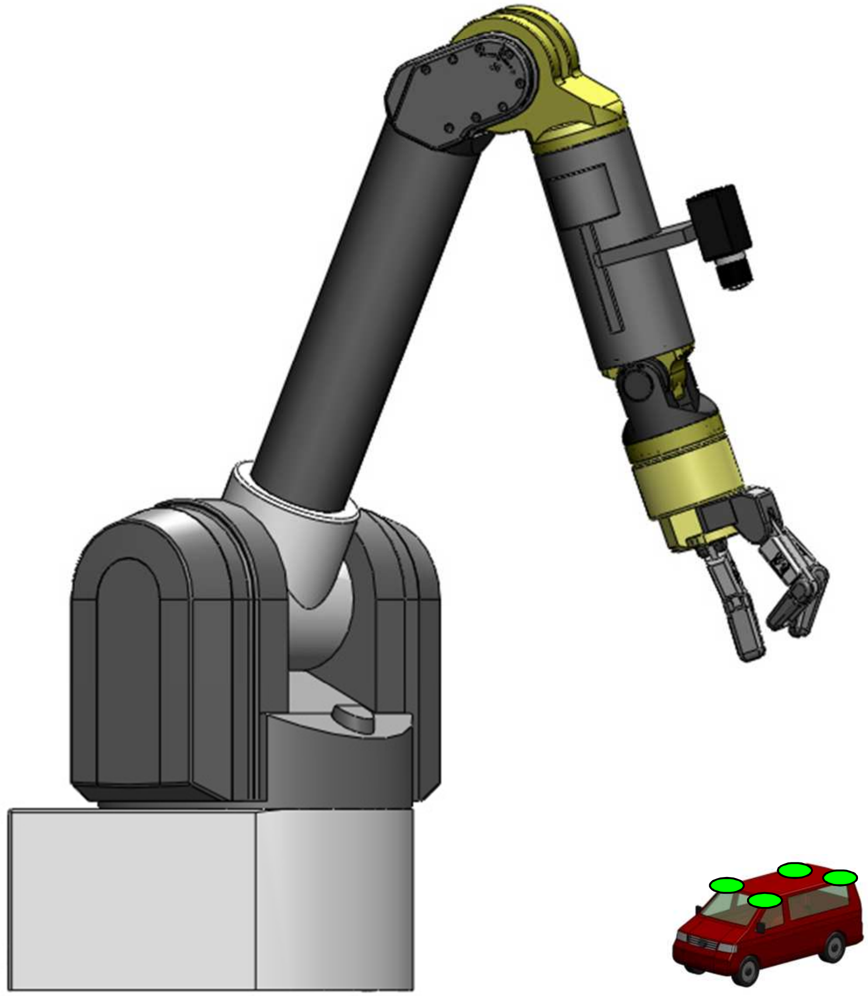
Vision-Based Control (Visual Servoing)

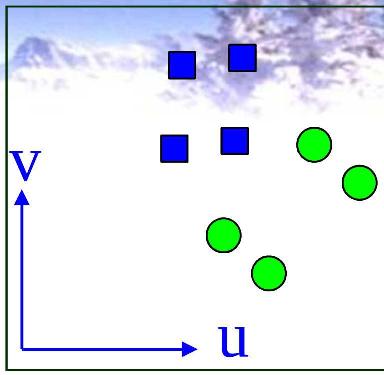




■ : Current Image Features
● : Desired Image Features

u,v Image-Space Error





■ : Current Image Features

● : Desired Image Features

$$\mathbf{E} = [\,\text{●} - \text{■}\,]$$

One point

$$\mathbf{E} = [\mathbf{y}_0 - \mathbf{y}^*]$$

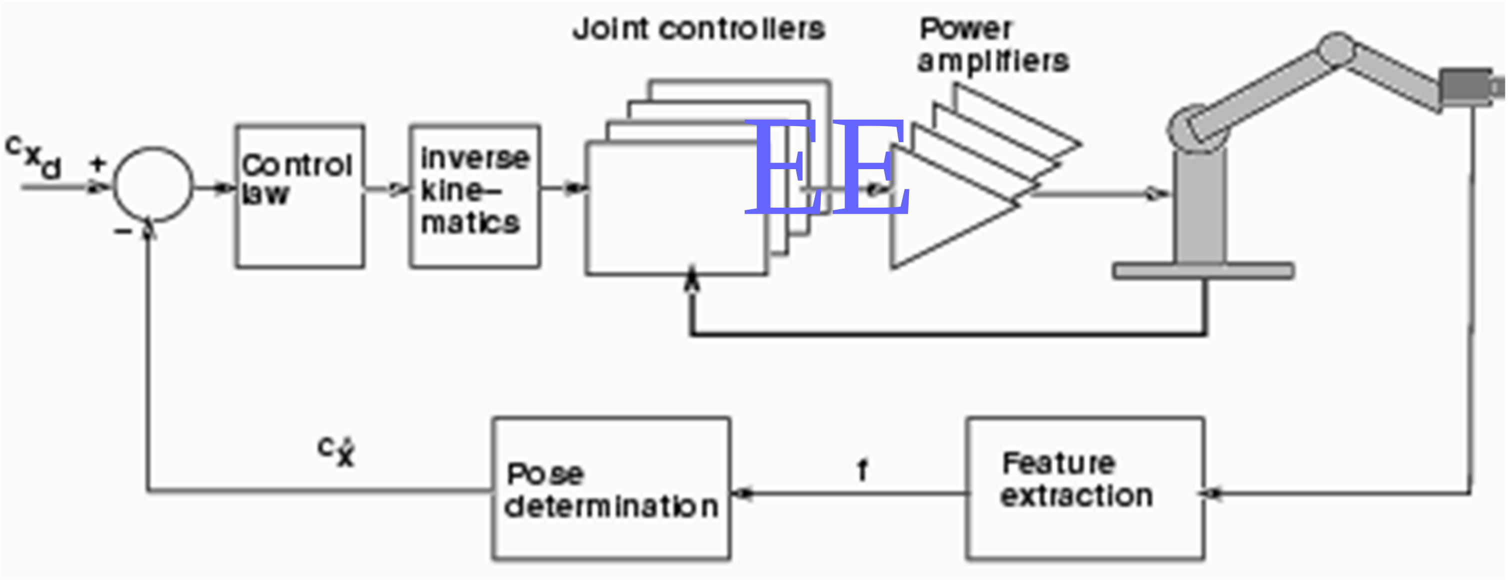
Pixel coord

$$\mathbf{E} = \begin{bmatrix} y_u \\ y_v \end{bmatrix} - \begin{bmatrix} y_u \\ y_v \end{bmatrix}^*$$

Many points

$$\mathbf{E} = \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}^* - \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}_0$$

Conventional Robotics: Motion command in base coord



- We focus on the geometric transforms

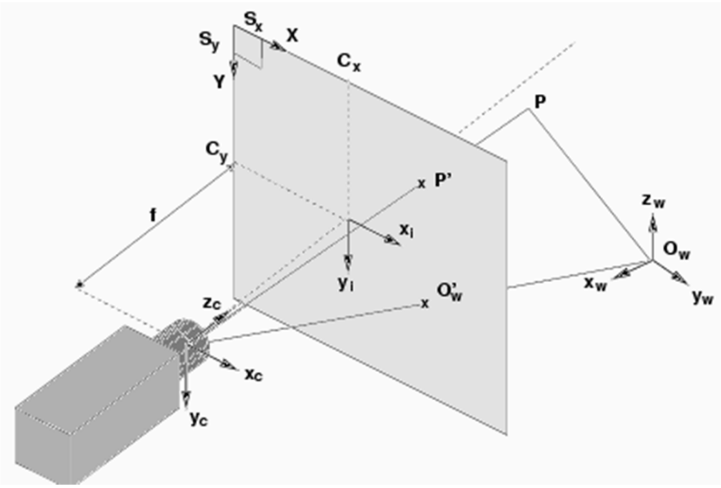
Problem: Lots of coordinate frames to calibrate

Camera

- Center of projection
- Different models

Robot

- Base frame
- End-effector frame
- Object



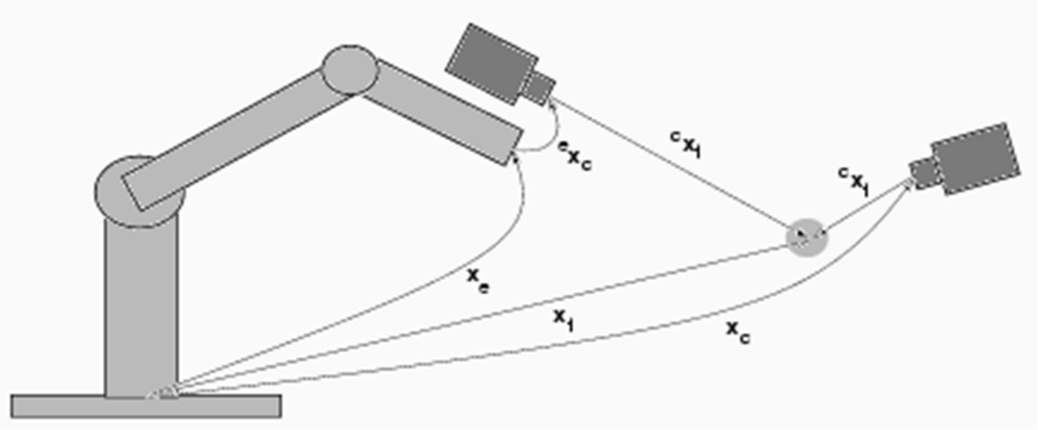
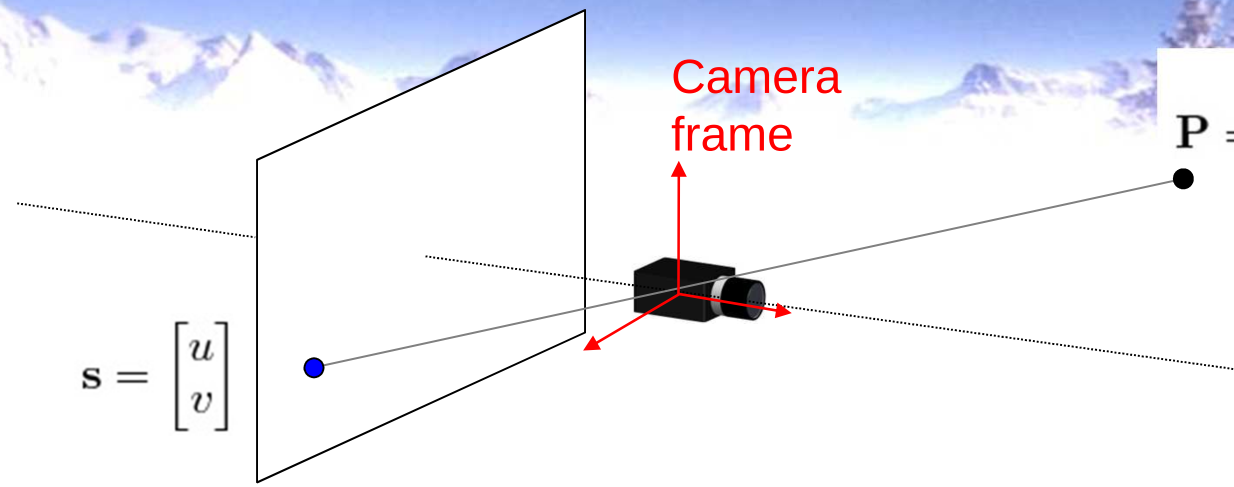


Image Formation is Nonlinear

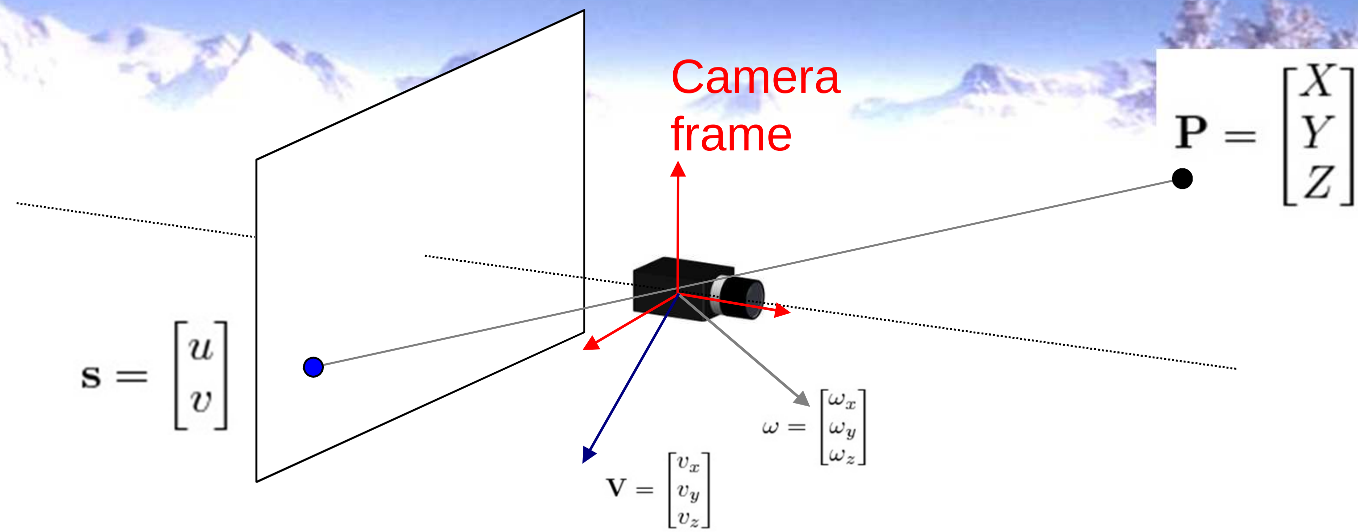


$$\mathbf{s} = \begin{bmatrix} u \\ v \end{bmatrix} \qquad \mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$u = f\frac{X}{Z}$$

$$v = f\frac{Y}{Z}$$

Camera Motion to Feature Motion



$$u = f\frac{X}{Z} \qquad v = f\frac{Y}{Z}$$

$$\Rightarrow \begin{cases} \dot{u} = f\left(\frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2}\right) = f\frac{\dot{X}}{Z} - \frac{u\dot{Z}}{Z} \\ \dot{v} = f\left(\frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2}\right) = f\frac{\dot{Y}}{Z} - \frac{v\dot{Z}}{Z} \end{cases}$$

$$\mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \Rightarrow \dot{\mathbf{P}} = -\mathbf{V} - \omega \times \mathbf{P} \quad \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases}$$

Camera Motion to Feature Motion

$$\begin{cases} \dot{u} = f\dfrac{-v_x - \omega_y Z + \omega_z Y}{Z} - \dfrac{u(-v_z - \omega_x Y + \omega_y X)}{Z} \\ \dot{v} = f\dfrac{-v_y - \omega_z X + \omega_x Z}{Z} - \dfrac{v(-v_z - \omega_x Y + \omega_y X)}{Z} \end{cases}$$

$$\begin{cases} \dot{u} = -\dfrac{f}{Z}v_x + 0.v_y + \dfrac{u}{Z}v_z + \dfrac{uv}{f}\omega_x - (f + \dfrac{u^2}{f})\omega_y + v\omega_z \\ \dot{v} = 0.v_x - \dfrac{f}{Z}v_y + \dfrac{v}{Z}v_z + (f + \dfrac{v^2}{f})\omega_x - \dfrac{uv}{f}\omega_y - u\omega_z \end{cases}$$

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

Interaction matrix is a Jacobian.

$$\dot{\mathbf{s}} = \mathbf{L}\begin{bmatrix} \mathbf{V} \\ \omega \end{bmatrix}$$

This derivation is for one point.

Q. How to extend to more points?

Q. How many points needed?

Problem: Lots of coordinate frames to calibrate

Camera

– Center of projection

– Different models

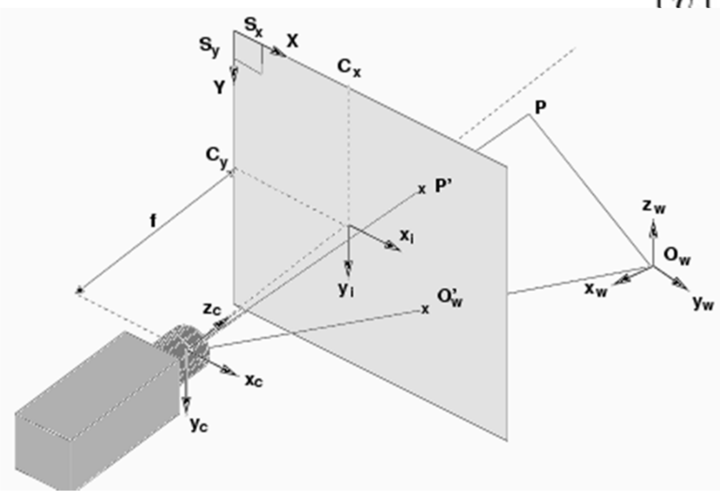
Robot

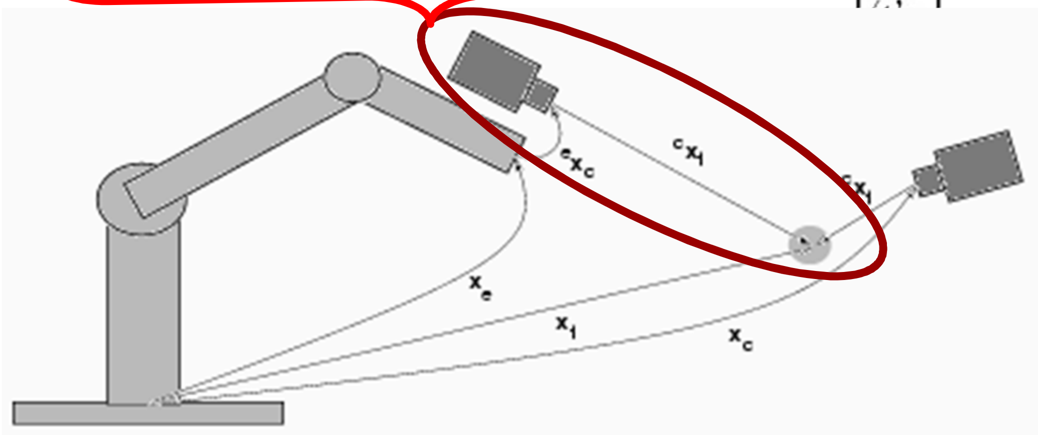
– Base frame

– End-effector frame

– Object

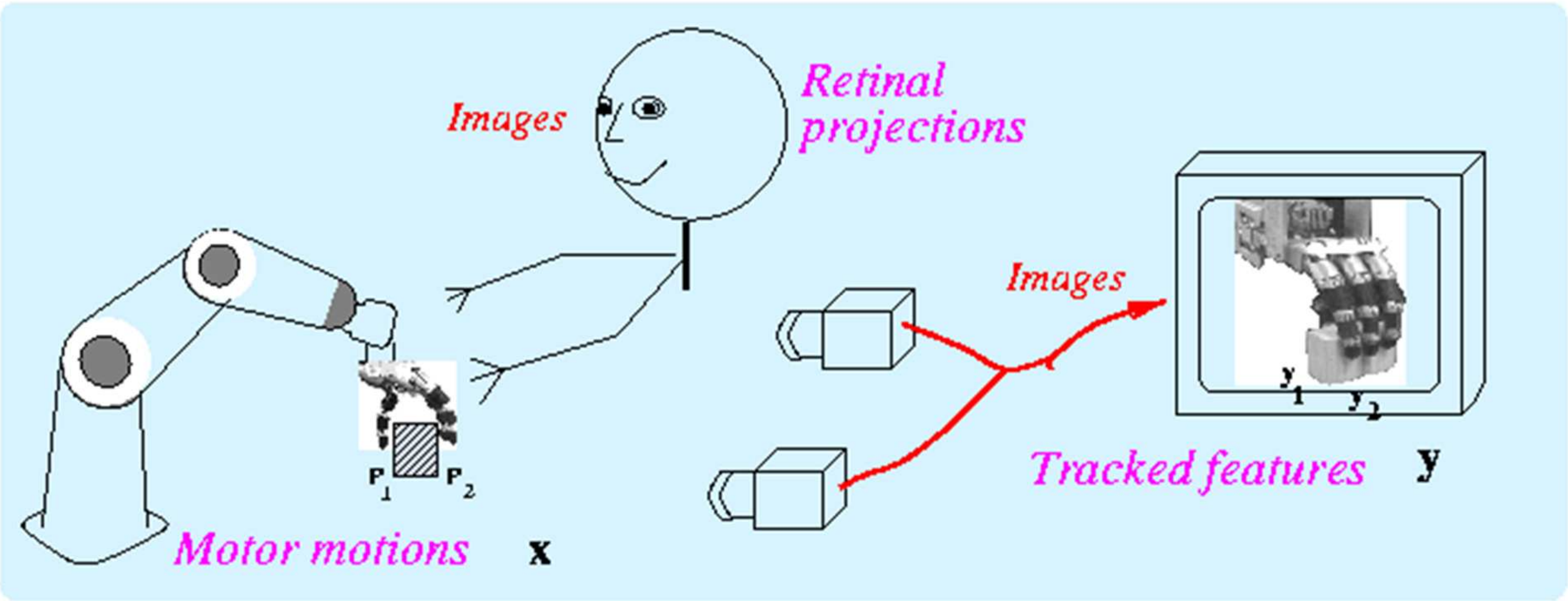
$$\begin{bmatrix}\dot{u}\\ \dot{v}\end{bmatrix} = \begin{bmatrix} -\frac{f}{Z} & 0 & \frac{u}{Z} & \frac{uv}{f} & -\frac{u^2+f^2}{f} & v \\ 0 & -\frac{f}{Z} & \frac{v}{Z} & \frac{v^2+f^2}{f} & -\frac{uv}{f} & -u \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$





Only covered one transform!

Other (easier) solution: Image-based motion control



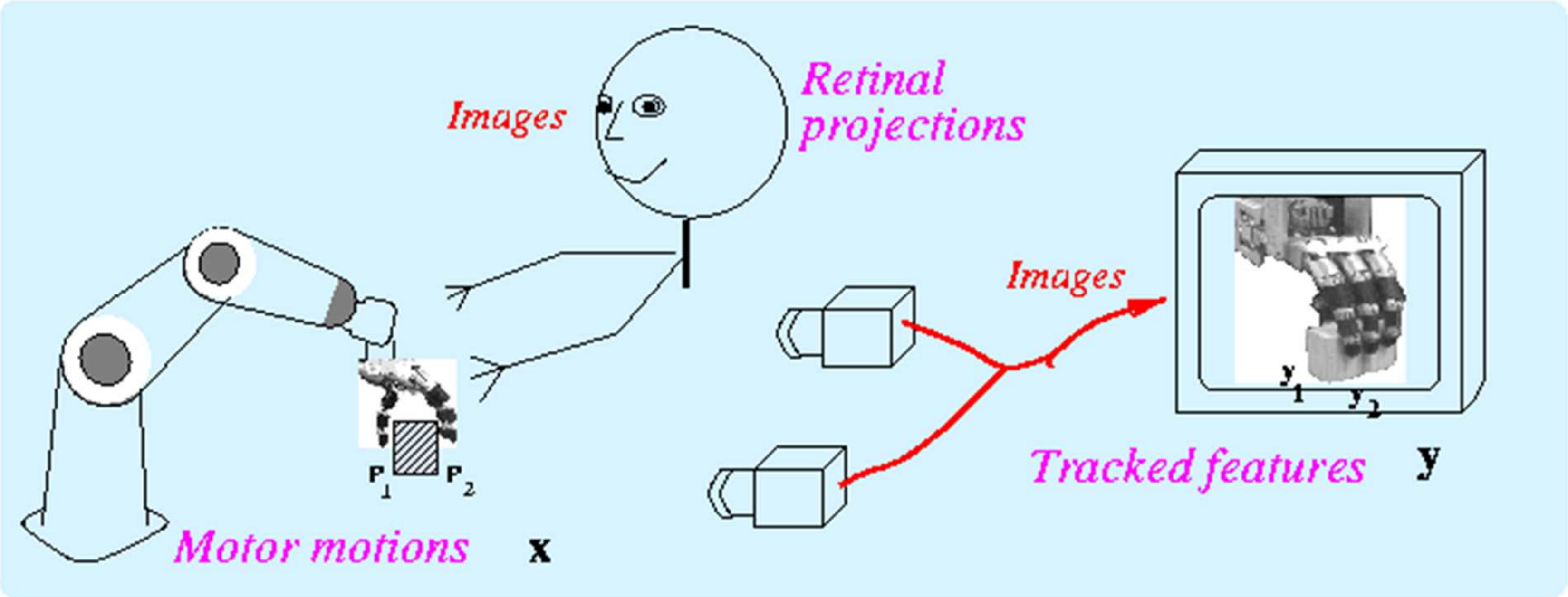
Motor-Visual function: y=f(x)

Achieving 3d tasks
via
2d image control

Other (easier) solution: Image-based motion control

Note: What follows will work for one or two (or 3..n) cameras.
Either fixed or on the robot.

Here we will use two cam



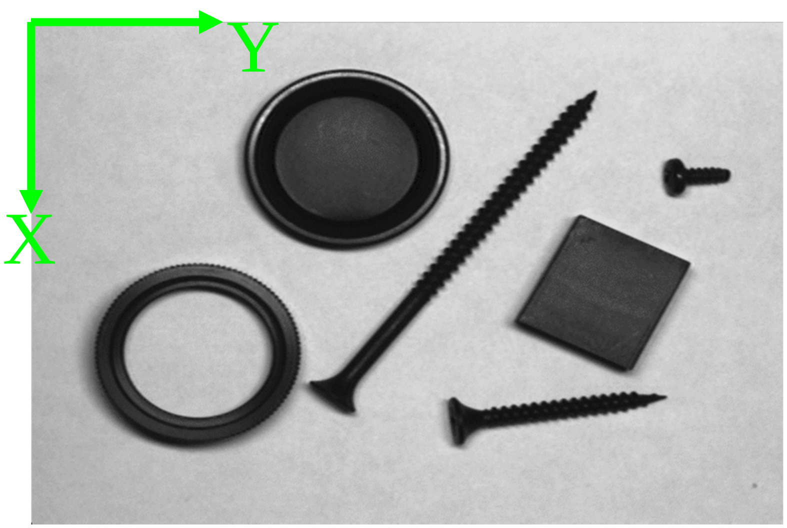
Motor-Visual function: y=f(x)

Achieving 3d tasks
via
2d image control

Simplest case: 2D planar world and aligned camera

This method is commonly used in industry machine vision

- Usually **overhead camera pointing straight down on a work table**
- Adjust cam position so pixel $[u,v] = s[X,Y]$. s = scalefactor (pix/mm)
- Pixel coordinates are scaled world coord

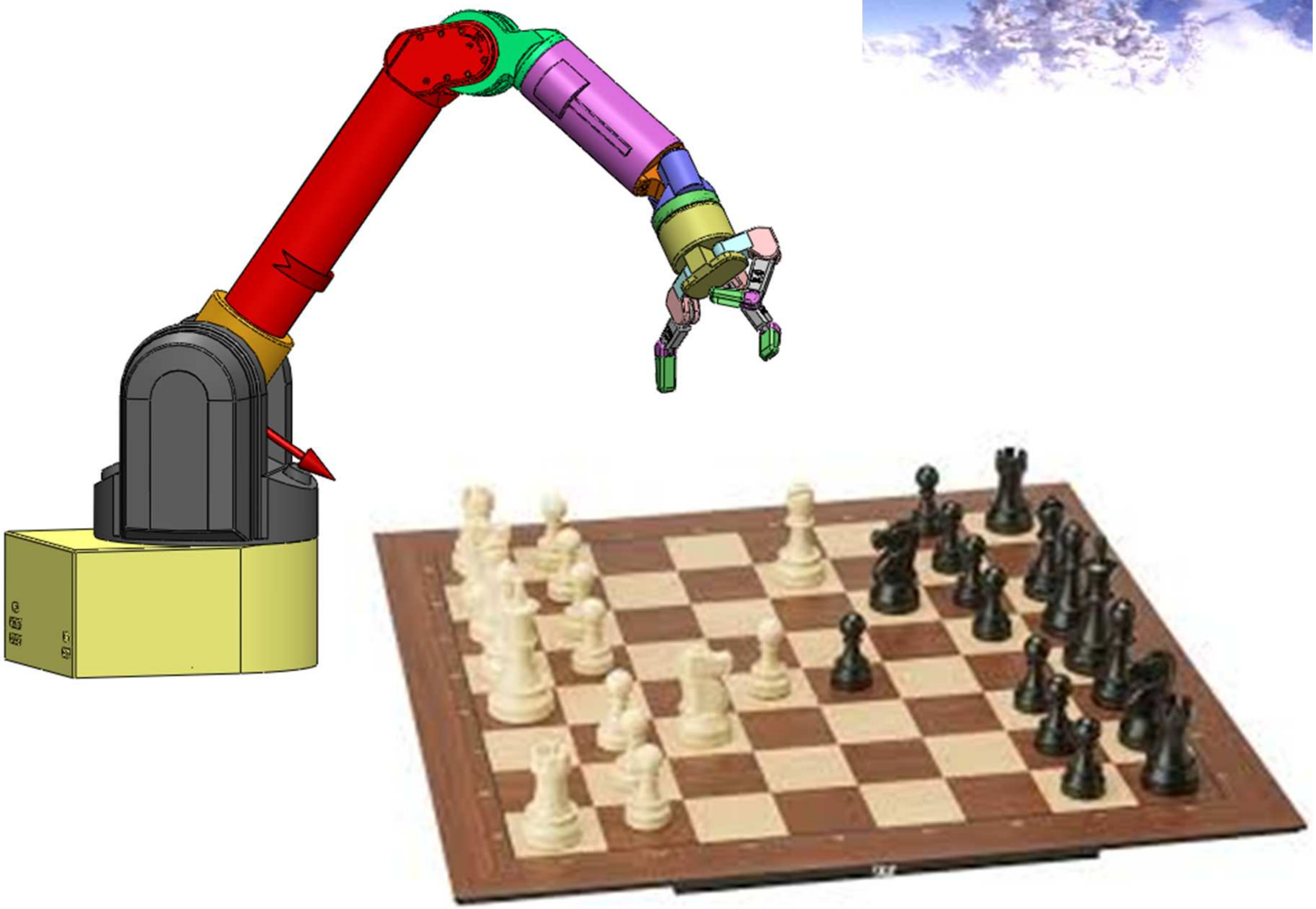


Robot scene

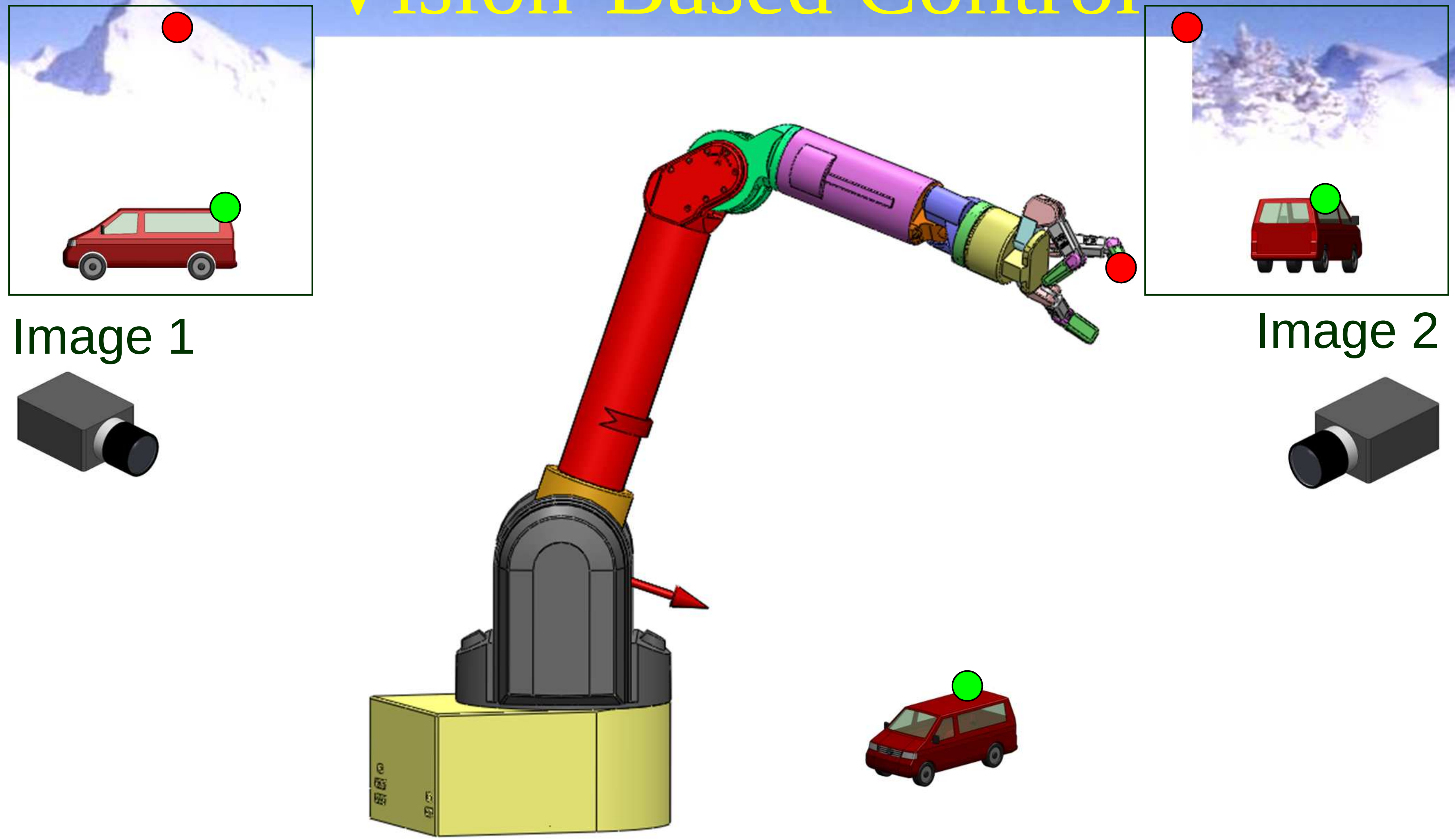
Camera

Thresholded image

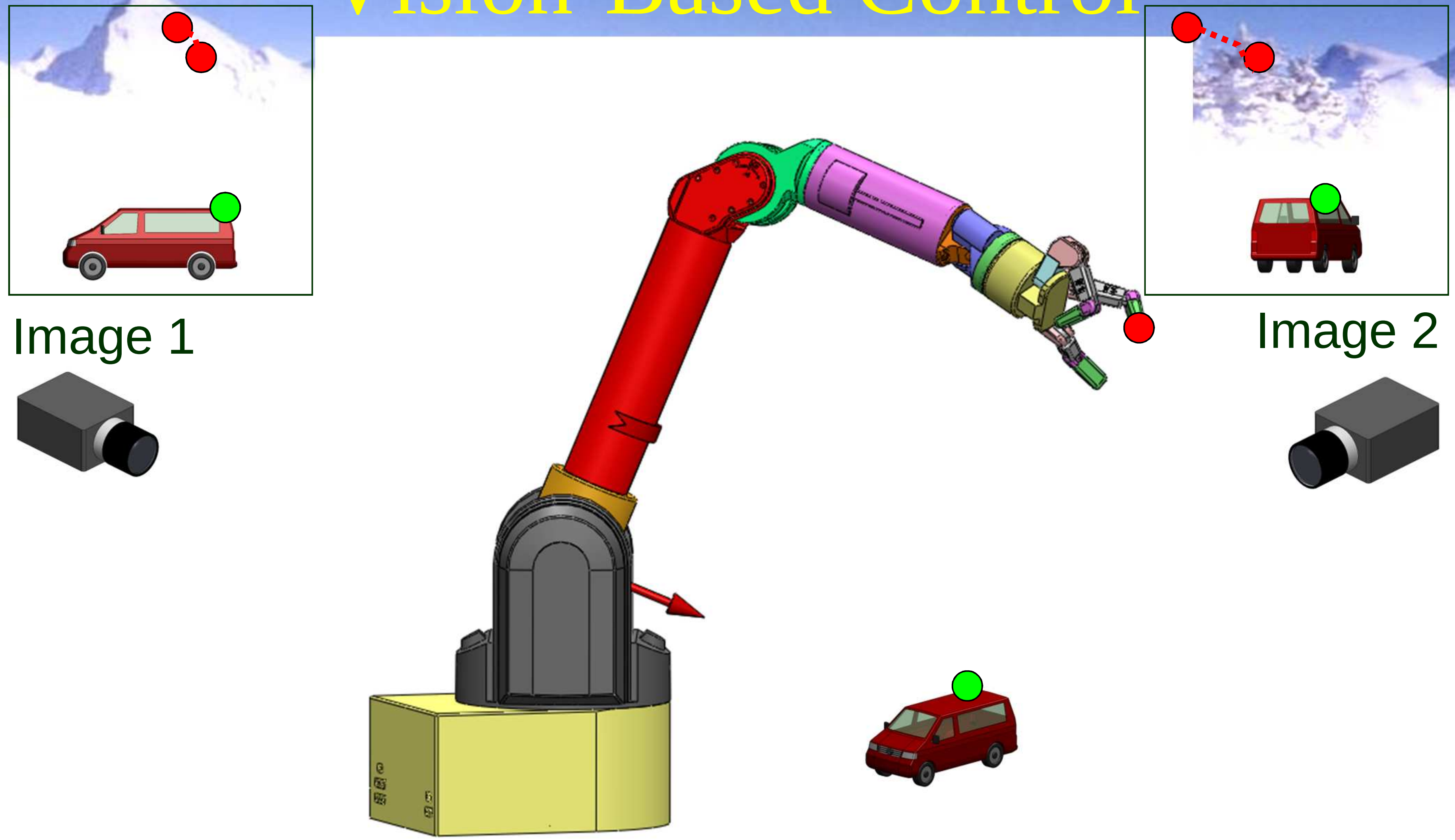
My first robotics problem



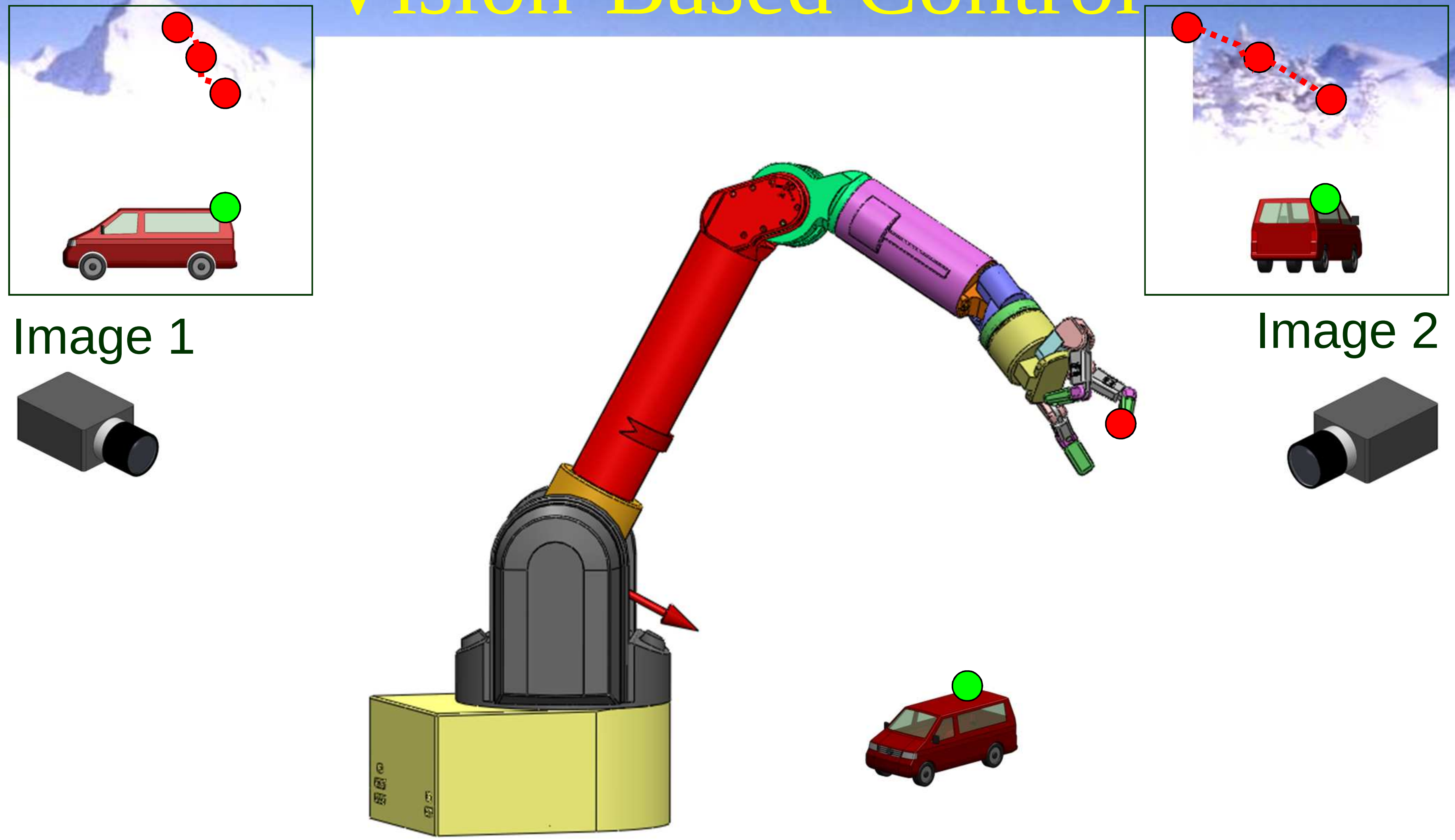
Vision-Based Control



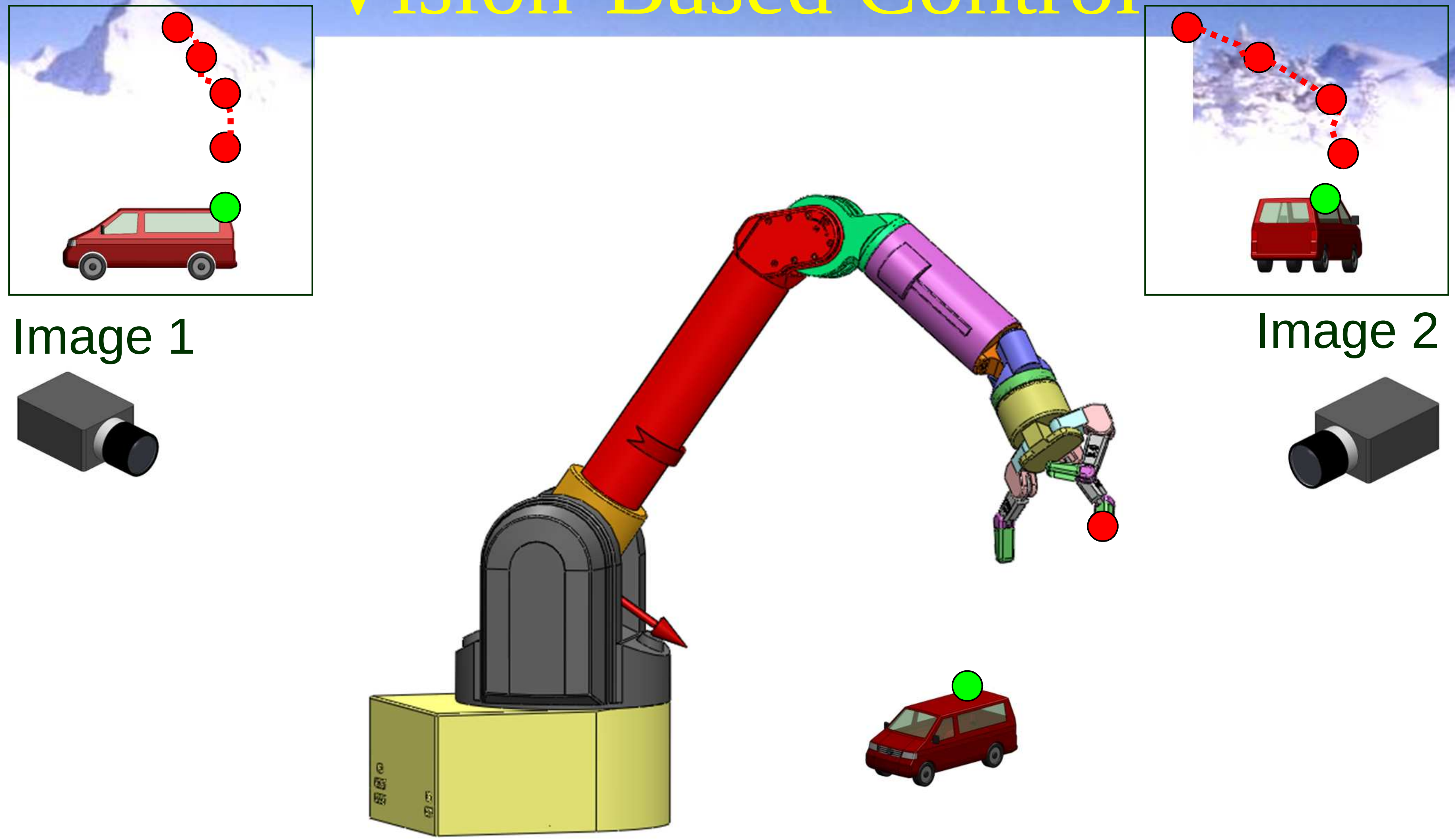
Vision-Based Control



Vision-Based Control



Vision-Based Control



Vision-Based Control

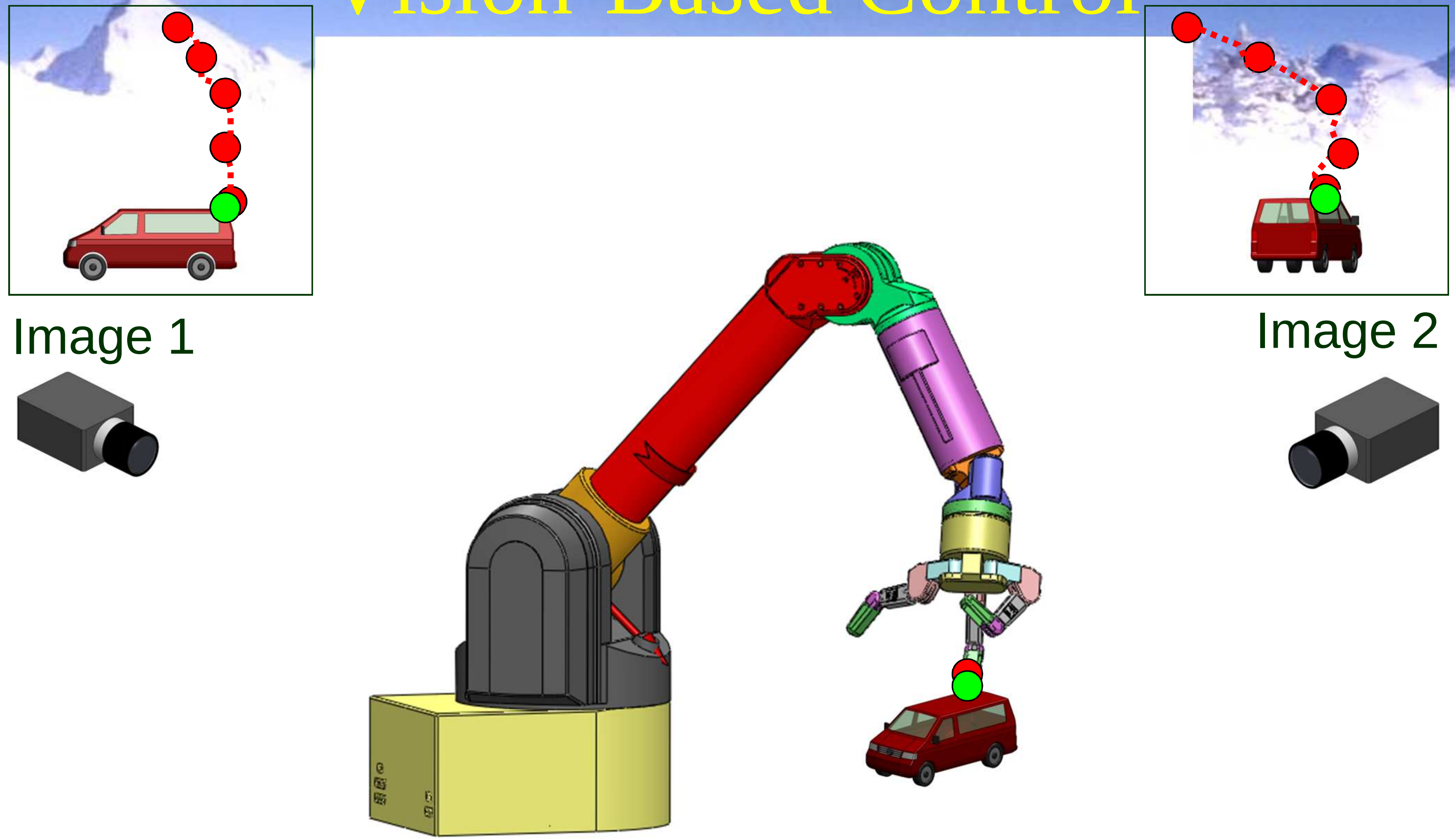
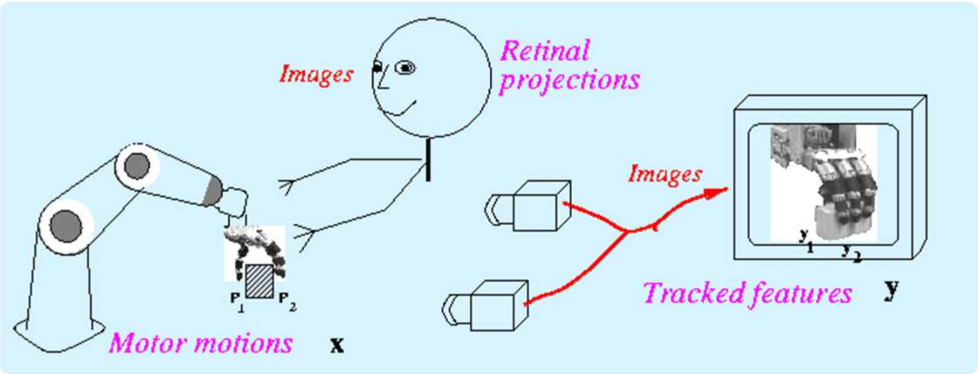


Image-based Visual Servoing

- Observed features: $\mathbf{y} = [\, y_1 \quad y_2 \quad \ldots \quad y_m \,]^T$
- Motor joint angles: $\mathbf{x} = [\, x_1 \quad x_2 \ldots \quad x_n \,]^T$
- Local linear model: $\Delta \mathbf{y} = \mathbf{J} \Delta \mathbf{x}$
- Visual servoing steps:
  1 Solve: $\mathbf{y}^* - \mathbf{y}_k = \mathbf{J} \Delta \mathbf{x}$
  2 Update: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}$



Find J Method 1:
Test movements along basis

- Remember: J is unknown m by n matrix

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \mathrm{f}_1}{\partial x_1} & \cdots & \frac{\partial \mathrm{f}_1}{\partial x_n} \\ \vdots & \ddots & \\ \frac{\partial \mathrm{f}_m}{\partial x_1} & & \frac{\partial \mathrm{f}_\mathrm{m}}{\partial x_n} \end{pmatrix}$$

- Motor moves (scale these):

$$\begin{aligned} \Delta \mathbf{x}_1 &= [1, 0, \ldots, 0]^T \\ \Delta \mathbf{x}_2 &= [0, 1, \ldots, 0]^T \\ &\vdots \\ \Delta \mathbf{x}_n &= [0, 0, \ldots, 1]^T \end{aligned}$$

- Suitable $10 < \Delta \mathbf{y} < 20$ pixels
- Finite difference:

$$\mathbf{J} \approx \left(\begin{bmatrix} \vdots \\ \Delta \mathbf{y}_1 \\ \vdots \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta \mathbf{y}_2 \\ \vdots \end{bmatrix} \cdots \begin{bmatrix} \vdots \\ \Delta \mathbf{y}_n \\ \vdots \end{bmatrix} \right)$$

Find J Method 2: Secant Constraints

- Constraint along a line:
- Defines m equations

$$\Delta \mathbf{y} = \mathbf{J} \Delta \mathbf{x}$$

- Collect n arbitrary, but different measures y
- Solve for J

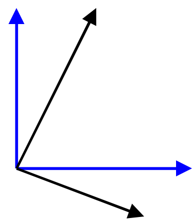
$$\begin{pmatrix} \begin{bmatrix} \cdots & \Delta \mathbf{y}_1^T & \cdots \end{bmatrix} \\ \begin{bmatrix} \cdots & \Delta \mathbf{y}_2^T & \cdots \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \cdots & \Delta \mathbf{y}_n^T & \cdots \end{bmatrix} \end{pmatrix} = \begin{pmatrix} \begin{bmatrix} \cdots & \Delta \mathbf{x}_1^T & \cdots \end{bmatrix} \\ \begin{bmatrix} \cdots & \Delta \mathbf{x}_2^T & \cdots \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \cdots & \Delta \mathbf{x}_n^T & \cdots \end{bmatrix} \end{pmatrix} \mathbf{J}^T$$

Find J Method 3: Recursive Secant Constraints

- Based on initial J and one measure pair $\Delta \mathbf{y}, \Delta \mathbf{x}$
- Adjust J s.t. $\Delta \mathbf{y} = \mathbf{J}_{k+1} \Delta \mathbf{x}$
- Rank 1 update:

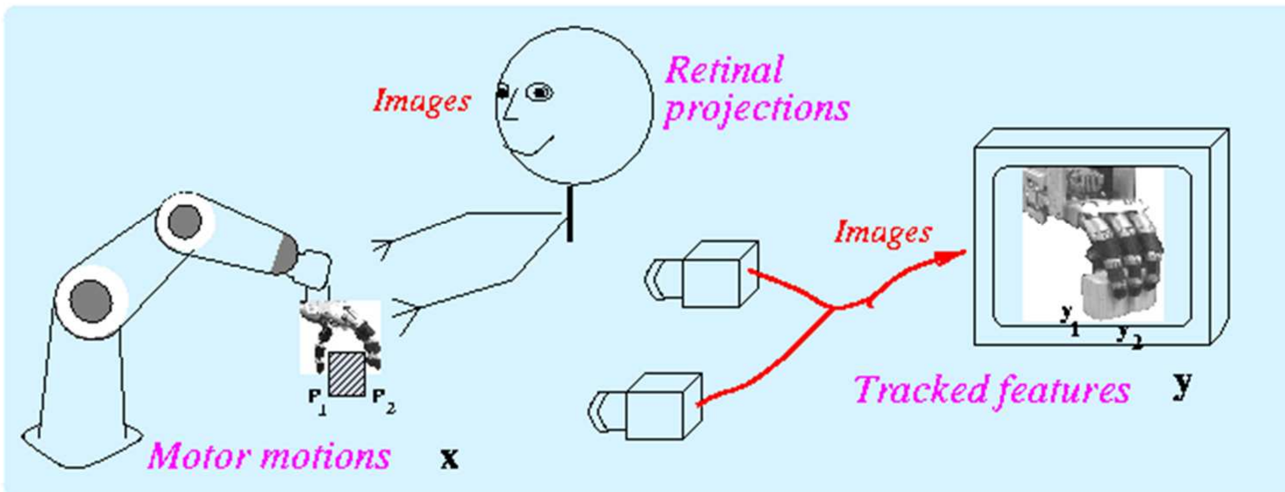
$$\hat{J}_{k+1} = \hat{J}_k + \frac{(\Delta \mathbf{y} - \hat{J}_k \Delta \mathbf{x}) \Delta \mathbf{x}^T}{\Delta \mathbf{x}^T \Delta \mathbf{x}}$$

- Consider rotated coordinates:



  - Update same as finite difference for n orthogonal moves

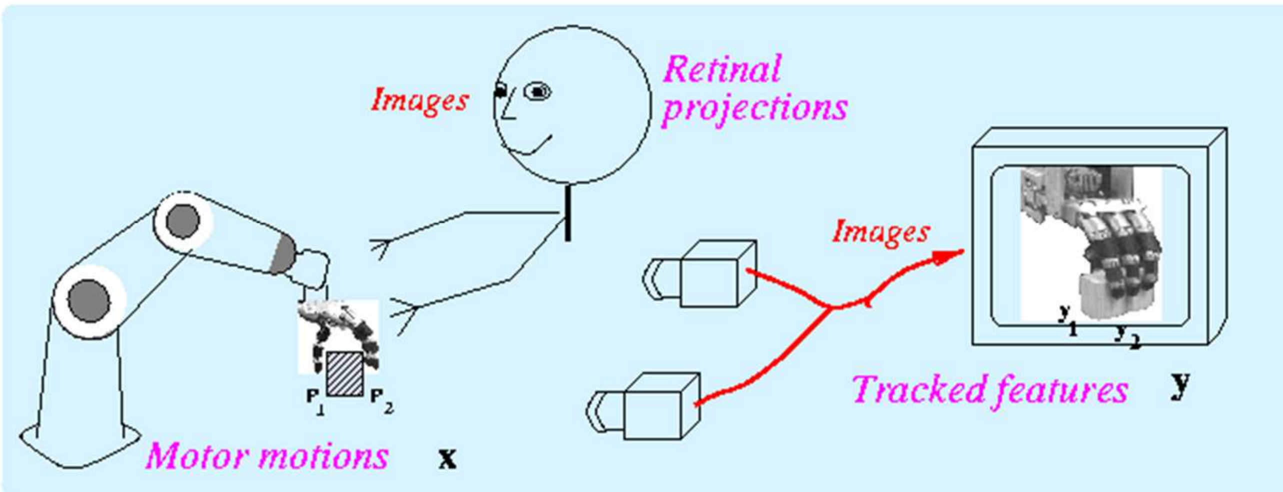
Achieving visual goals: Uncalibrated Visual Servoing



repeat

1. Solve for motion: $[\mathbf{y}^* - \mathbf{y}_k] = \mathbf{J}\Delta\mathbf{x}$
2. Move robot joints: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$
3. Read actual visual move Δy
4. Update Jacobian: $\hat{J}_{k+1} = \hat{J}_k + \frac{(\Delta\mathbf{y} - \hat{J}_k\Delta\mathbf{x})\Delta\mathbf{x}^T}{\Delta\mathbf{x}^T\Delta\mathbf{x}}$

Achieving visual goals: Uncalibrated Visual Servoing



r
e
p
e
a
t

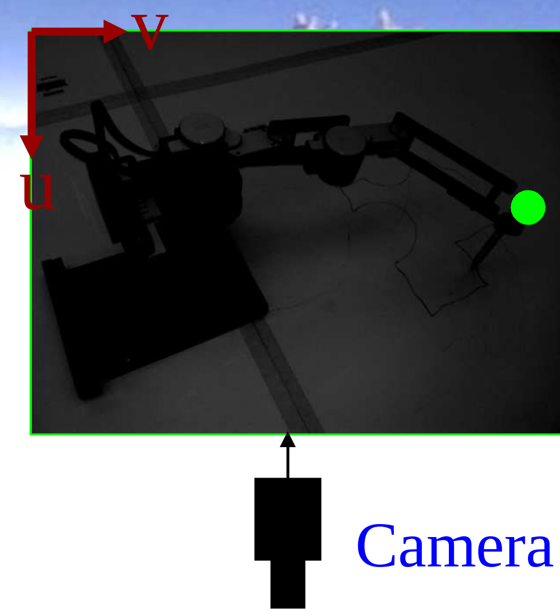
1. Solve for motion: $[\mathbf{y}^* - \mathbf{y}_k] = \mathbf{J}\Delta\mathbf{x}$
2. Move robot joints: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$
3. Read actual visual move Δy

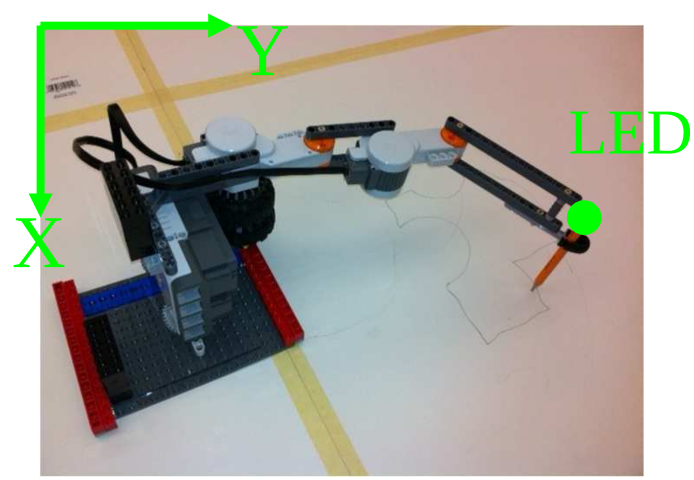
$$\frac{(\Delta \mathbf{y} - \hat{J}_k \Delta \mathbf{x})\Delta \mathbf{x}^T}{\Delta x^T \Delta x}$$

Can we always guarantee when a task is achieved/achievable?

What we need for VS

- Camera(s) Place for visibility and
- Tracking
  - Markerless algorithms - MTF
  - Put LED on robot end-effector
    - Lower brightness so LED brighterest
- Robot: Anywhere within reach
- Control command access
  - Joint posiiton move
  - Joint velocity
  - Cartesian
    - Issues with angle representation: Euler, quatern, exp



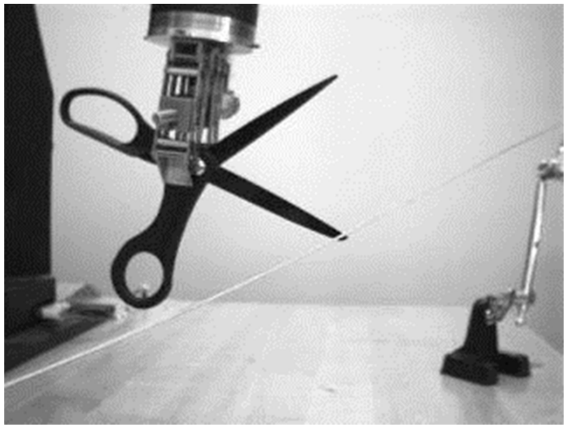


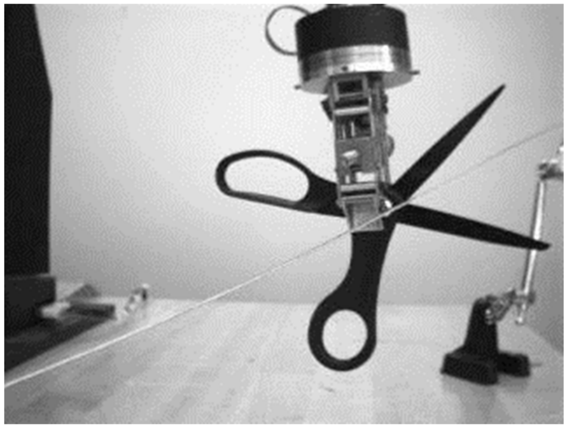
Visually guided motion control

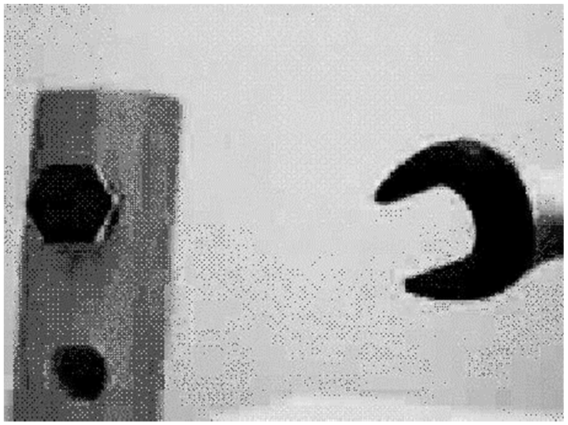
Issues:

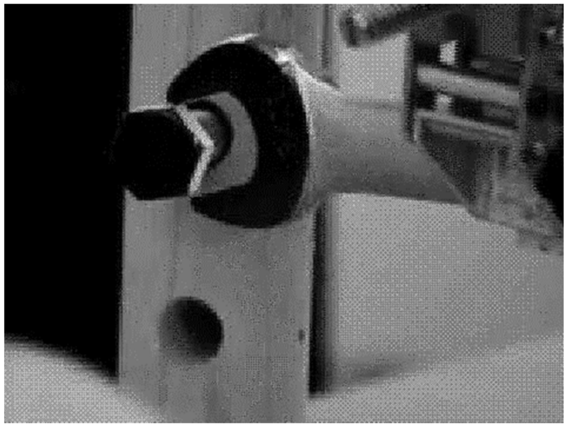
1. What tasks can be performed?
   – Camera models, geometry, visual encodings
2. How to do vision guided movement?
   – H-E transform estimation, feedback, feedforward motion control
3. How to plan, decompose and perform whole tasks?

How to specify a visual task?



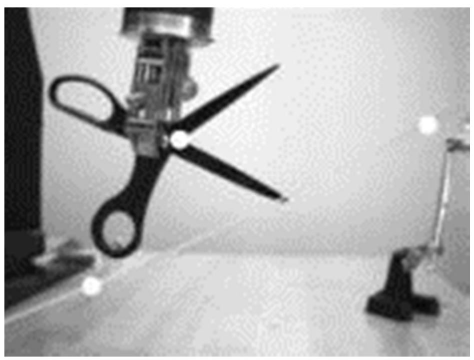






Task and Image Specifications

Task function $T(x) = 0$ Image encoding $E(y) = 0$



task space
error

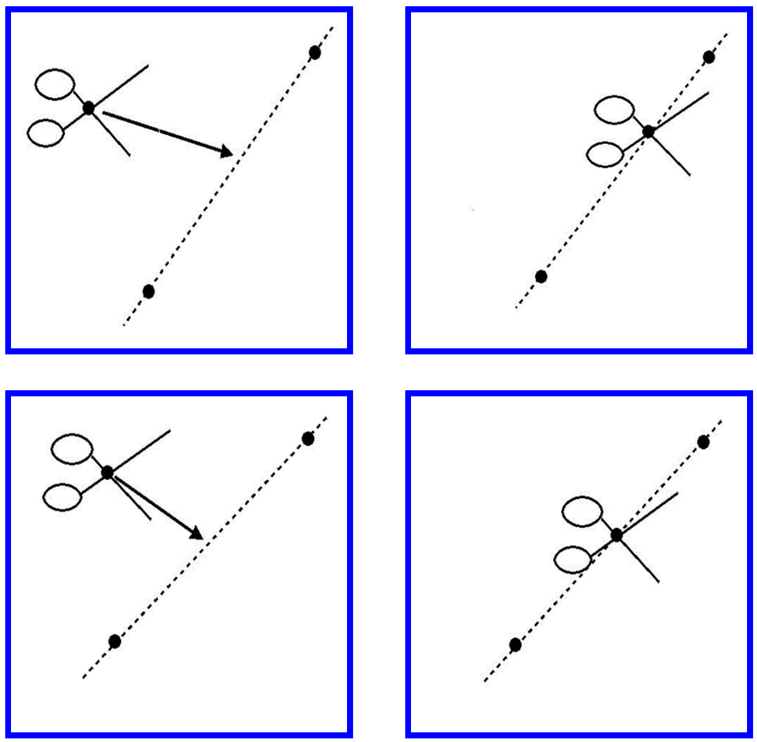
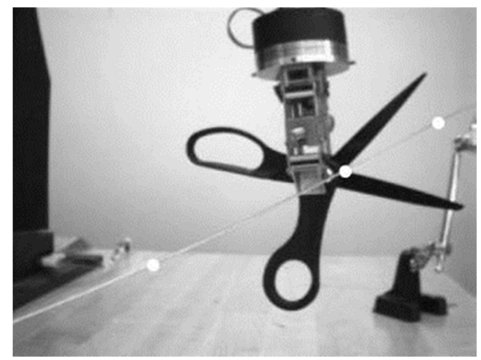


image space
error

image space
satisfied

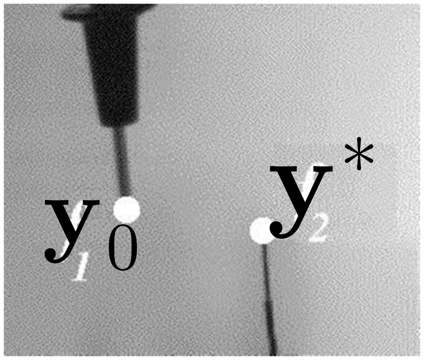


task space
satisfied

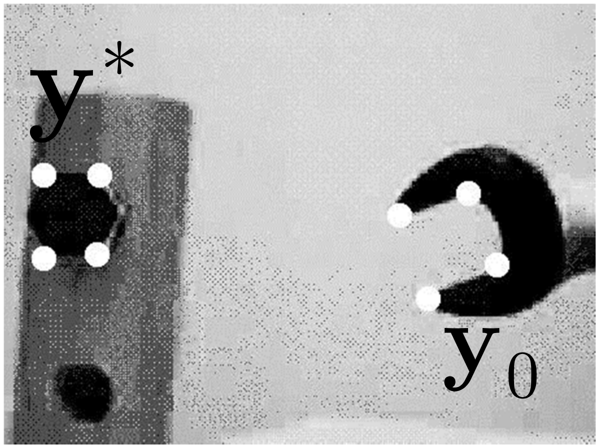
Visual specifications

- Point to Point task “error”:

$$\mathbf{E} = [\mathbf{y}_2^* - \mathbf{y}_0]$$



$$\mathbf{E} = \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}^* - \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}_0$$



Why 16 elements?

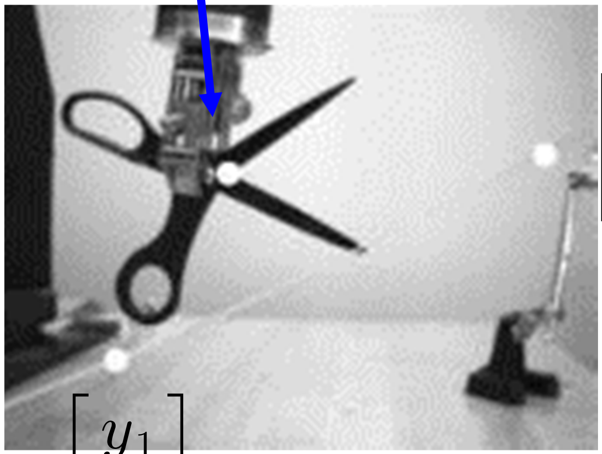
Visual specifications 2

- Point to Line

Line:

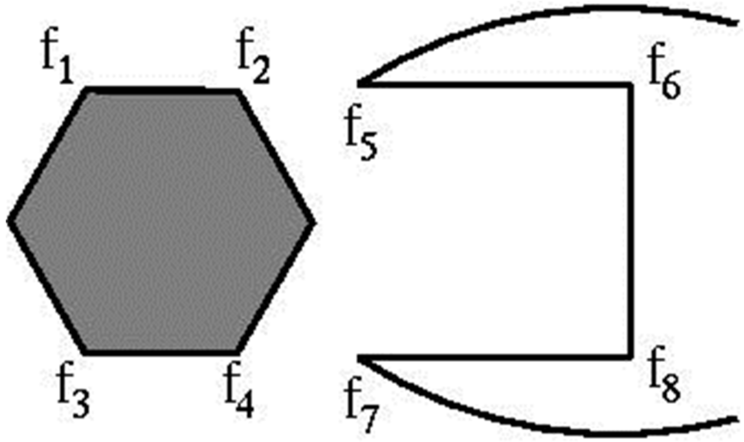
$$\mathbf{E}_{pl}(\mathbf{y},\mathbf{l}) = \begin{bmatrix} \mathbf{y}_l \cdot \mathbf{l}_l \\ \mathbf{y}_r \cdot \mathbf{l}_r \end{bmatrix}$$

$$\mathbf{l}_l = \begin{bmatrix} y_3 \times y_1 \\ y_4 \times y_2 \end{bmatrix}$$



Note: y homogeneous coord.

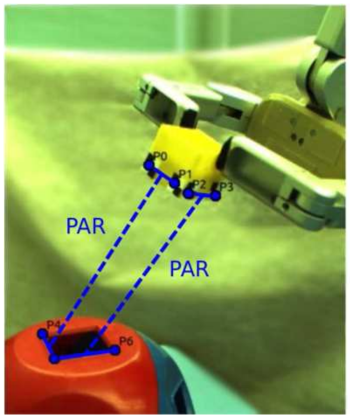
Parallel Composition example

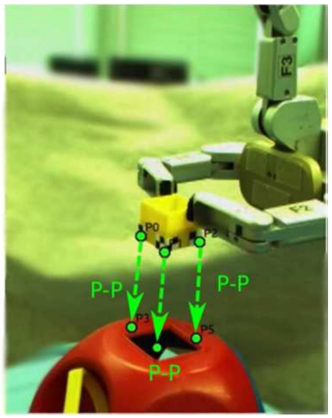


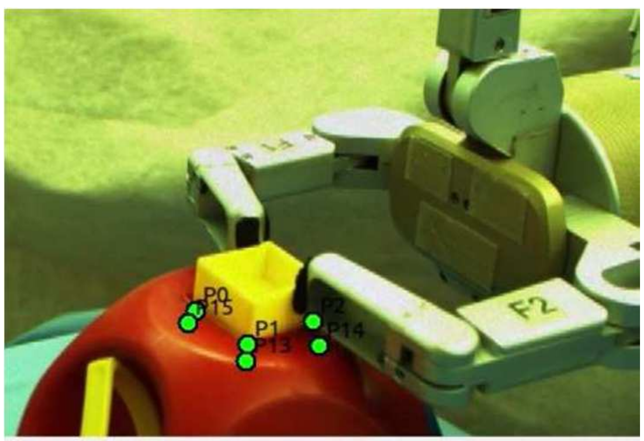
$$E_{wrench}(y) = \begin{bmatrix} y_2 - y_5 \\ y_4 - y_7 \\ y_6 \bullet (y_1 \times y_2) \\ y_8 \bullet (y_3 \times y_4) \end{bmatrix}$$

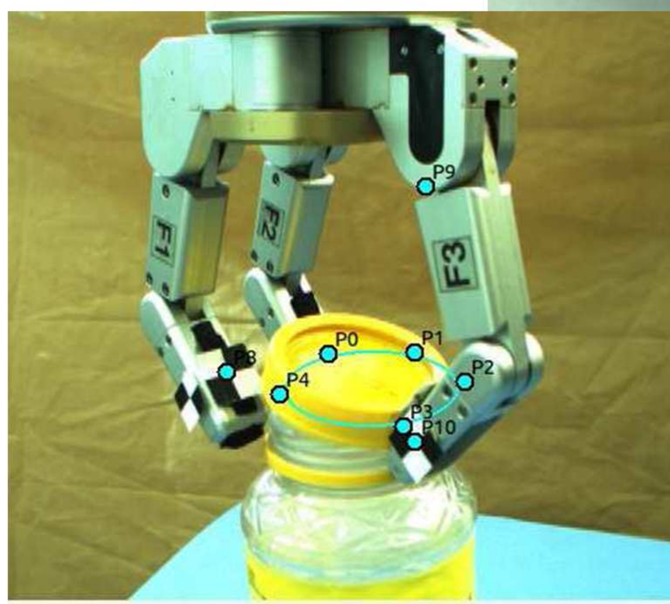
(plus e.p. checks)

How to define a visual task?



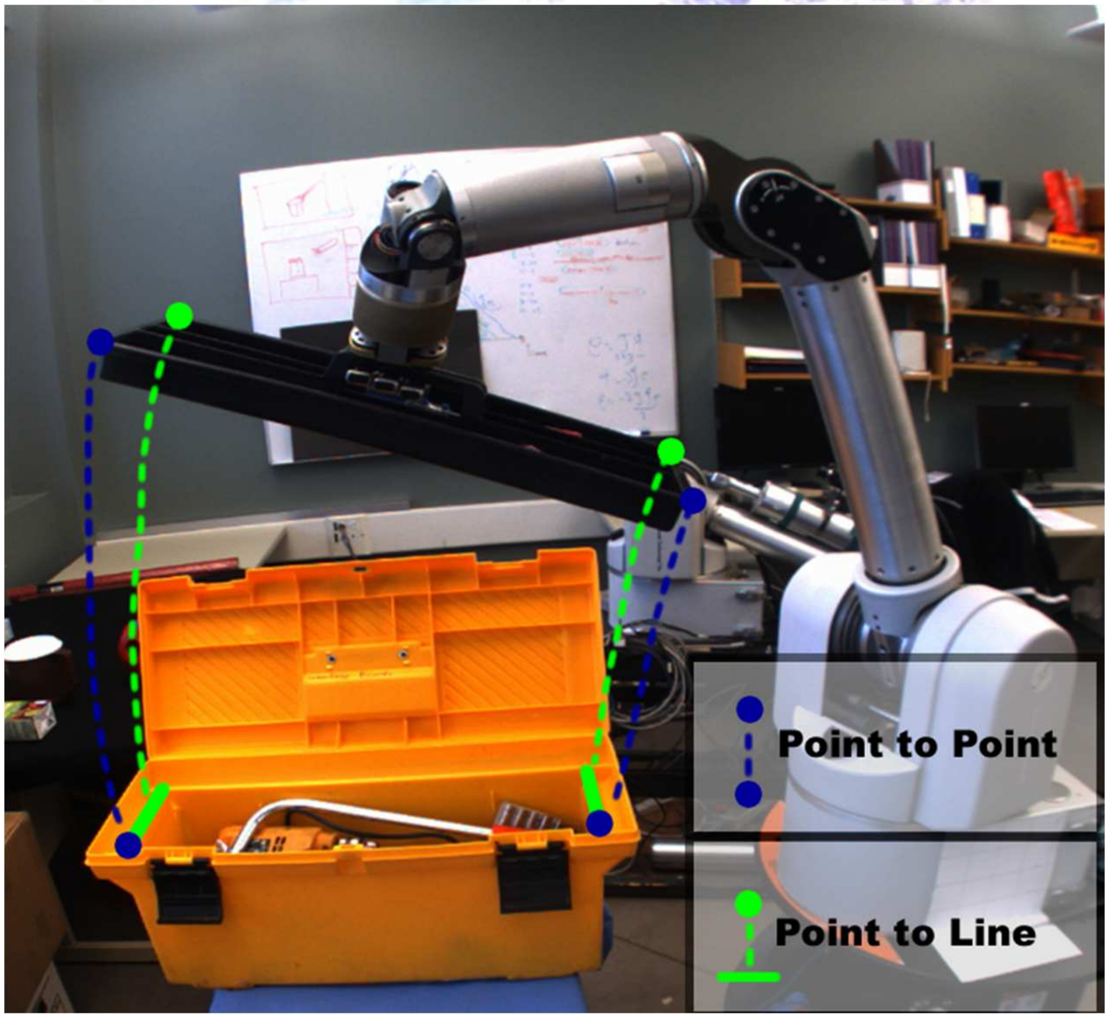






5.4. Visual specifications

- Image commands = Geometric constraints in image space
- HRI: User points/clicks on features in an image
- Robot controller drives visual error to zero



Visually defined alignment: basic

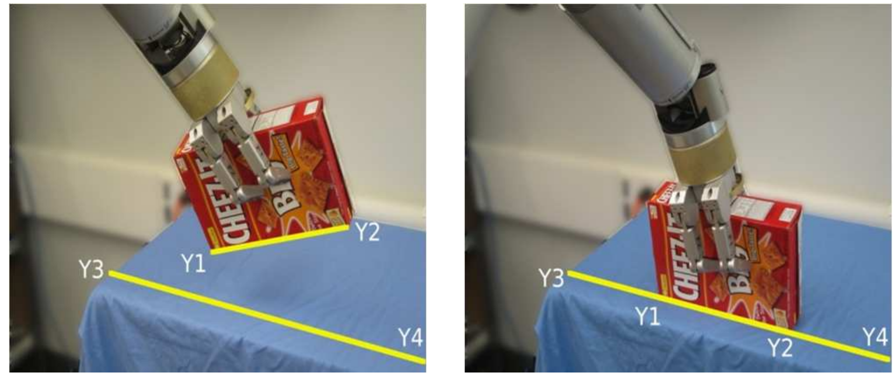
point-to-point: $e_{pp}(y) = y_2 - y_1$

or (homogenous coord) $e_{pp}(y) = y_2 \times y_1$

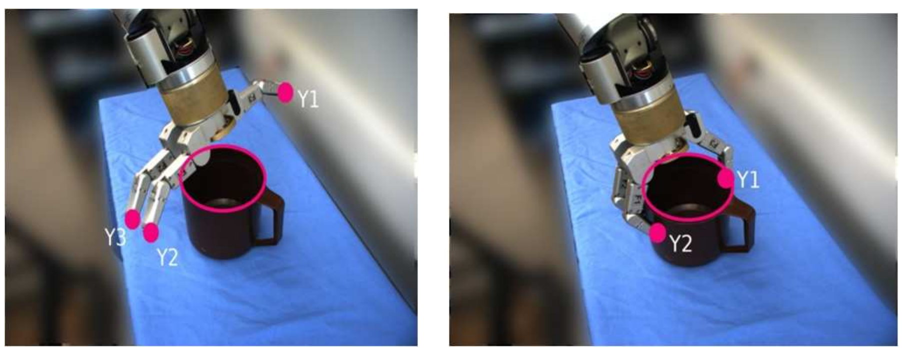
point-to-line: $e_{pl}(y) = y_1 \cdot (y_2 \times y_3)$

Some more visual alignments

parallel lines: $e_{par}(l) = (l_1 \text{ x } l_2) \text{ x } (l_3 \text{ x } l_4)$



line-to-line: $e_{ll}(y) = y_1 \cdot (y_3 \text{ x } y_4) + y_2 \cdot (y_3 \text{ x } y_4)$

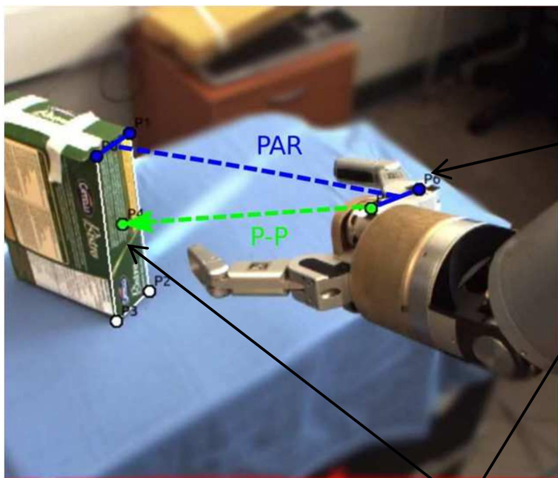


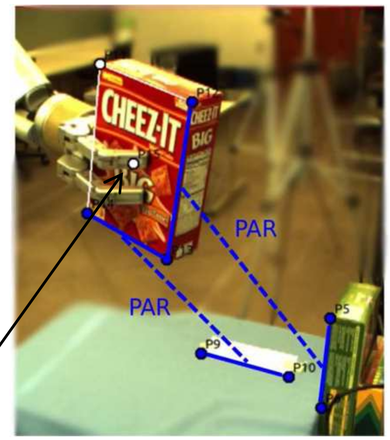
point-to-ellipse: $e_{pe}(y) = y^T{}_1 \, C_{ellipse} \, y_1$

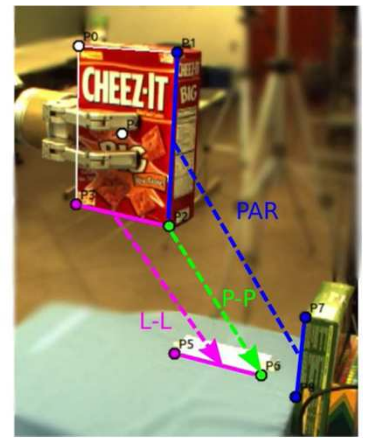
Now: Language for visual alignments What else do we need?

Need: 1. Some way of entering alignments in images

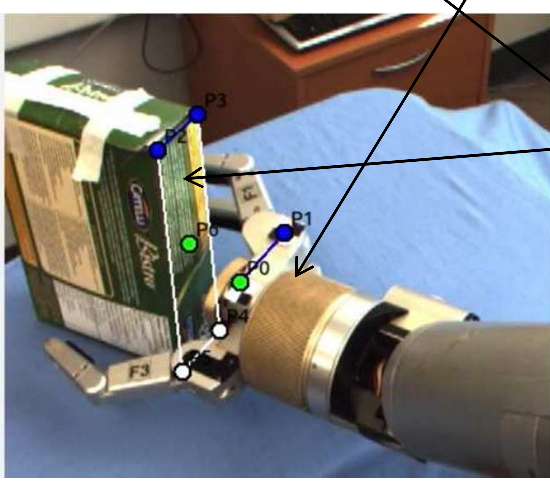
2. video tracking to perform servoing!



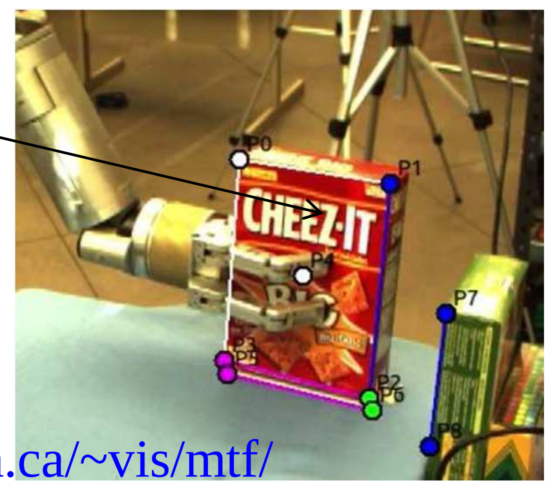




Feature trackers



Registration trackers:

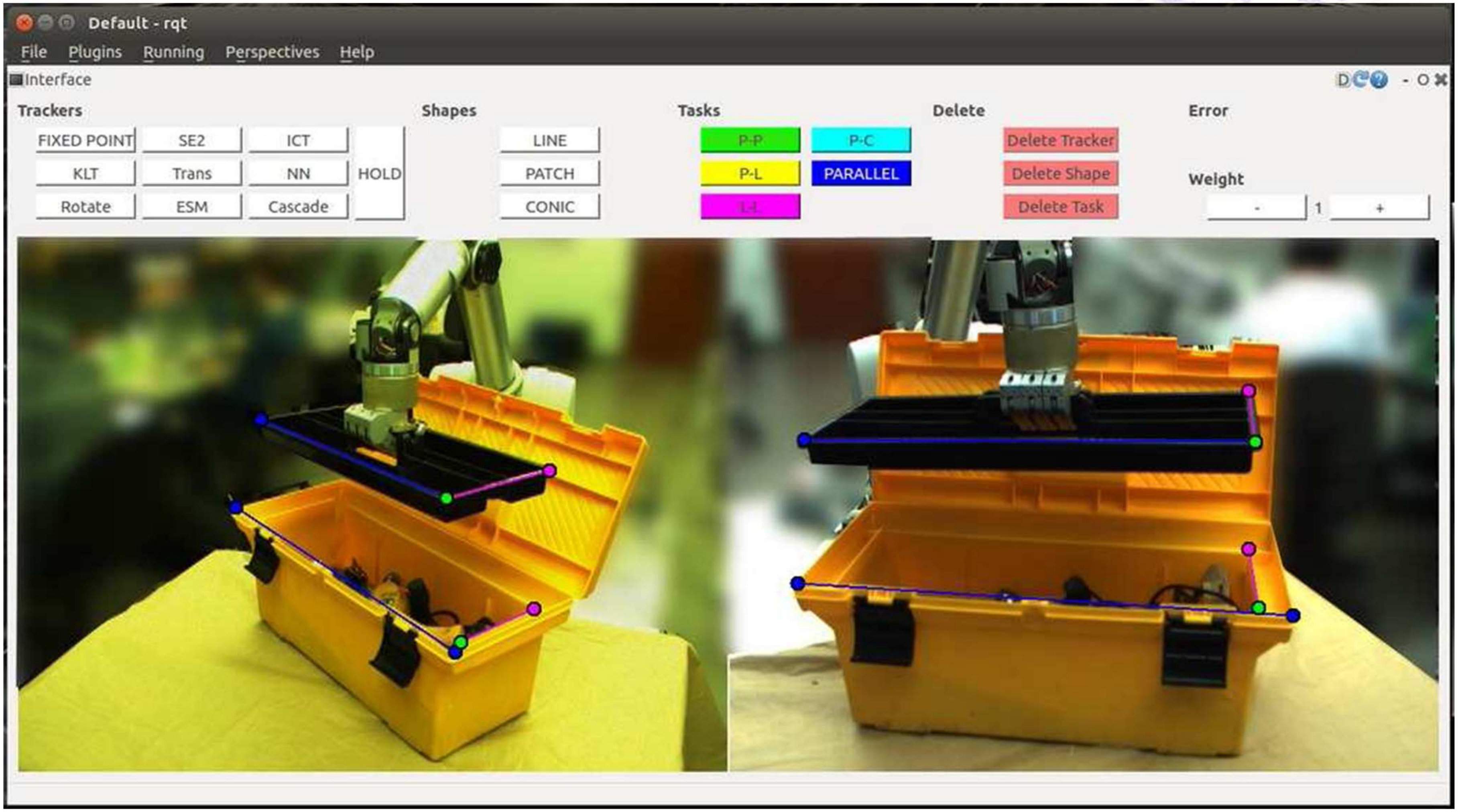


Download:

http://webdocs.cs.ualberta.ca/~vis/mtf/

ViTa: Visual Task spec. for Manip.

Mona Gridseth, Martin Jagersand ICRA'16



Composing basic visual alignments
1. Parallel

Conceptual task

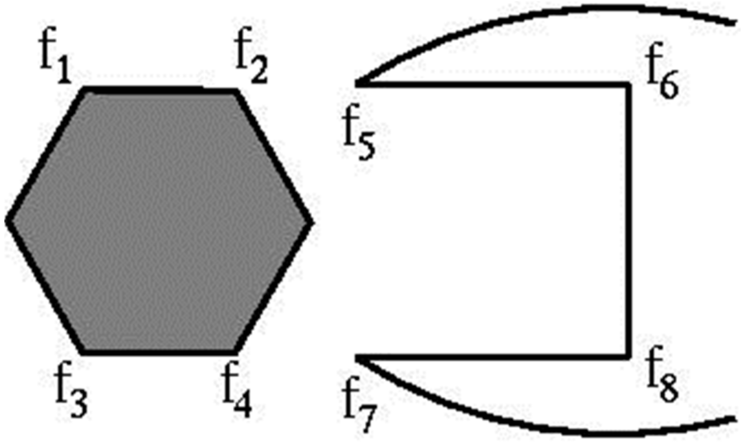
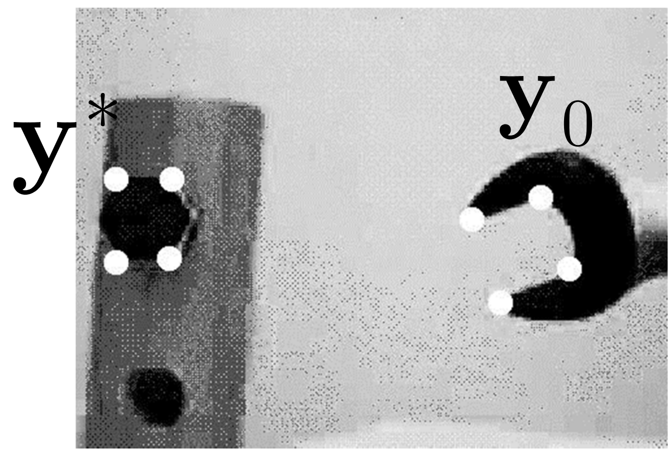


Image of task



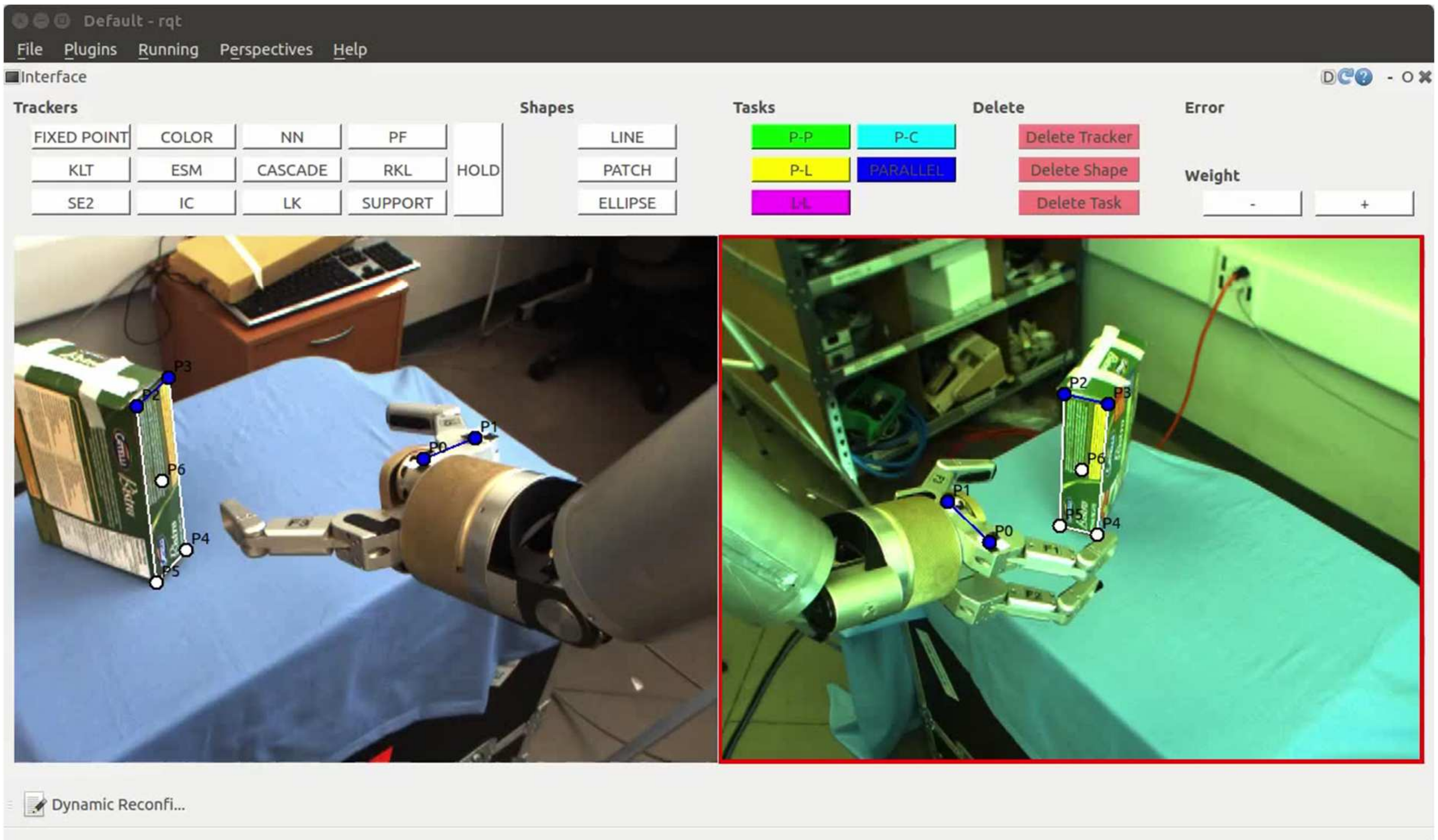
$$\mathbf{E} = \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}^* - \begin{bmatrix} y_1 \\ \vdots \\ y_{16} \end{bmatrix}_0$$

$$E_{wrench}(y) = \begin{pmatrix} y_2 - y_5 \\ y_4 - y_7 \\ y_6 \bullet (y_1 \times y_2) \\ y_8 \bullet (y_3 \times y_4) \end{pmatrix}$$

Which encoding E(y) is better? Can both reach E(y) = 0 ?

Do we need 1 or 2 cameras? Why?

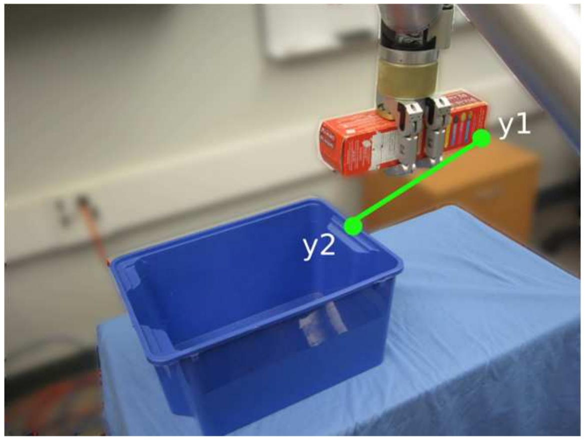
Composing basic visual alignments
1. Parallel constrains

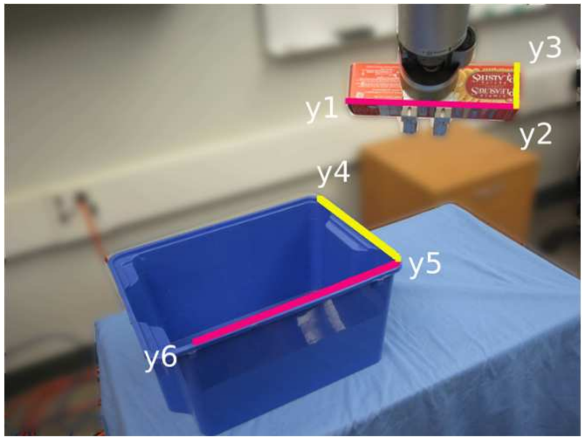


Composing basic visual alignments 2. Serial seqencing

Many tasks divide naturally into

1. Transportation / reaching
   - Coarse primitive for large movements
   - Low DOF control (e.g. of object centroid)
   - Robust to disturbances
2. Fine Manipulation
   - For high precision control of both position and orientation
   - 6DOF control based on several object features



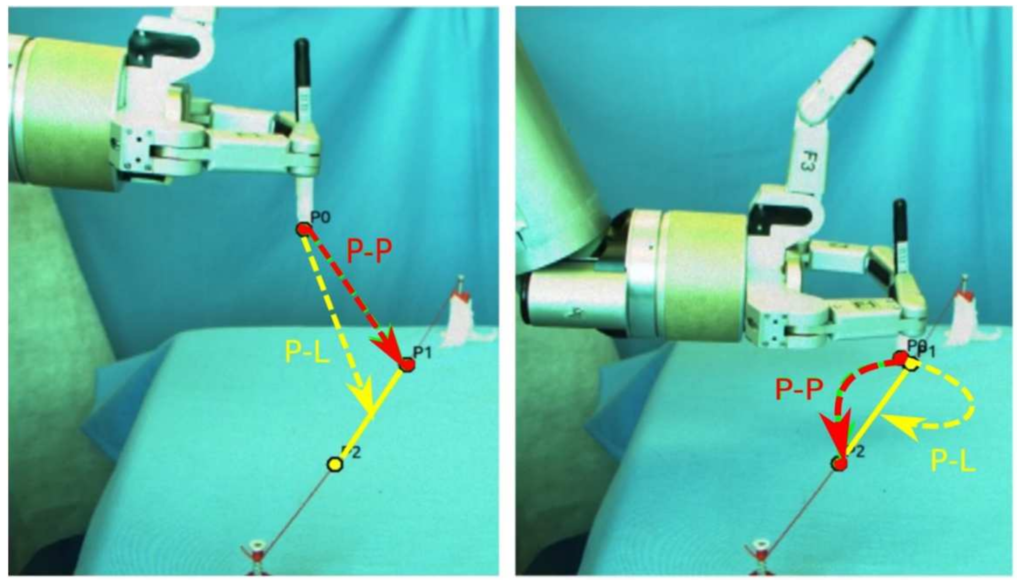


Serial Composition
Solving whole real tasks

- Linked task primitive

$$A = (\mathbf{E}^{\text{init}}, M, \mathbf{E}^{\text{final}})$$

1. Acceptable initial (visual) conditions
2. Visual or Motor constraints to be maintained
3. Final desired condition



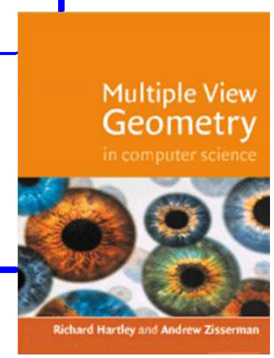
- Task = $A_1 A_2 \ldots A_k$

Previous alignments just examples. Can use any invariants for the level of calibration available

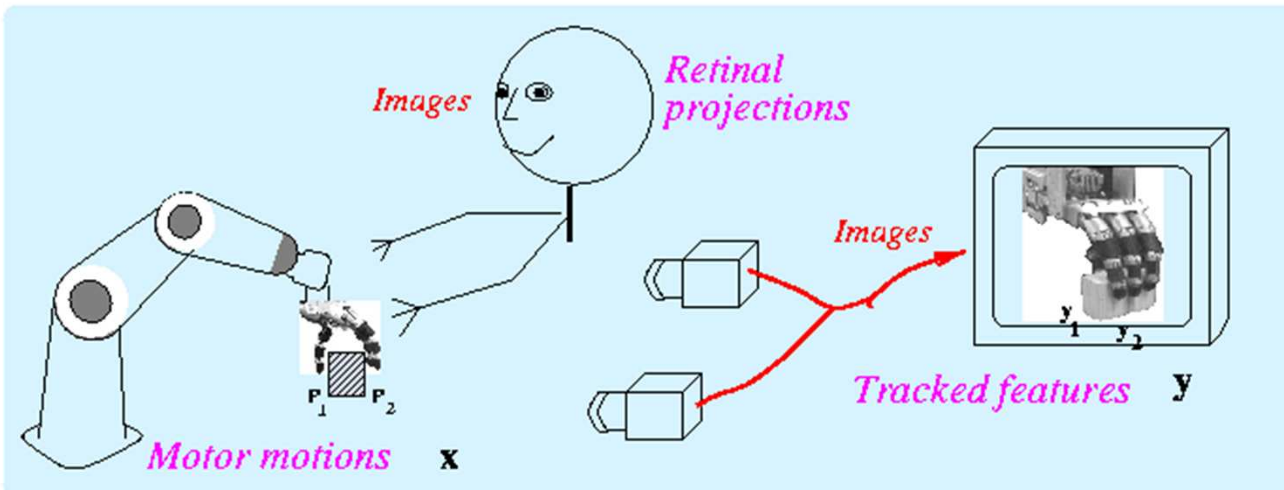
Group	Transformation	Invariants	Distortion
Projective 8 DOF	$H_P = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{v}^T & v \end{bmatrix}$	• Cross ratio • Intersection • Tangency	
Affine 6 DOF	$H_A = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	• Parallelism • Relative dist in 1d • Line at infinity $\mathbf{l}_\infty$	
Metric 4 DOF	$H_S = \begin{bmatrix} sR & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	• Relative distances • Angles • Dual conic C^*_∞	
Euclidean 3 DOF	$H_E = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$	• Lengths • Areas	

2 dof $\mathbf{l}_\infty$

2 dof C^*_∞



Achieving visual goals: Uncalibrated Visual Servoing



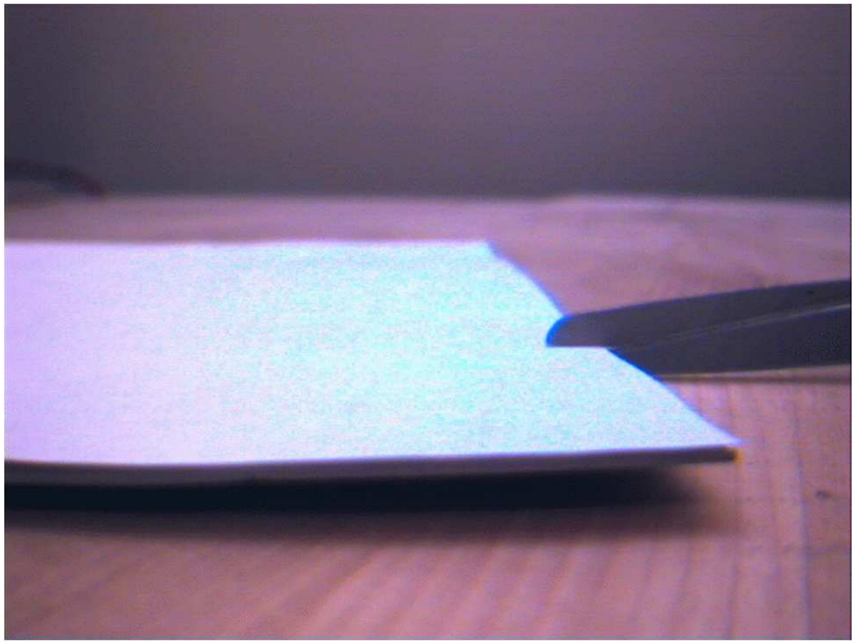
repeat

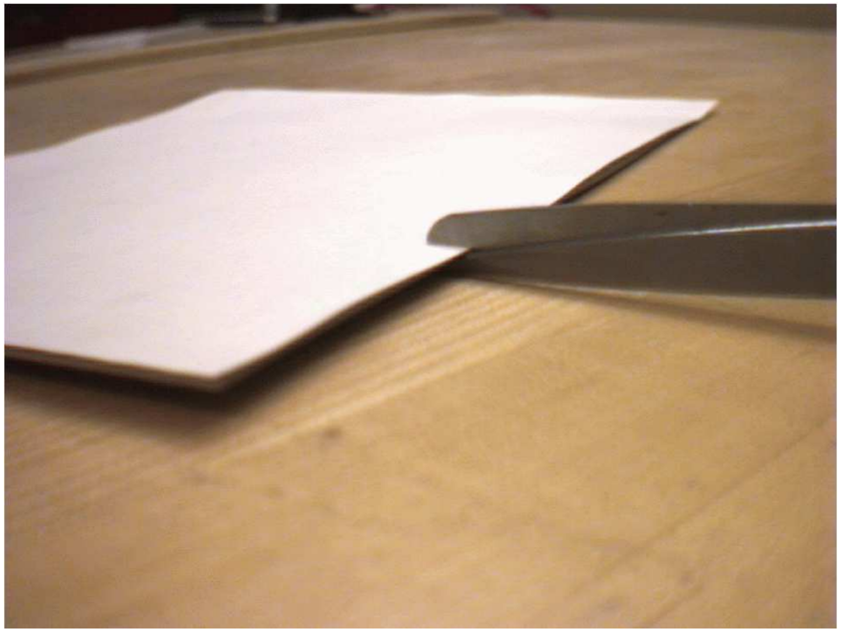
1. Solve for motion: $[\mathbf{y}^* - \mathbf{y}_k] = \mathbf{J}\Delta\mathbf{x}$
2. Move robot joints: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$
3. Read actual visual move Δy

$$\frac{(\Delta\mathbf{y} - \hat{J}_k \Delta\mathbf{x})\Delta\mathbf{x}^T}{\Delta x^T \Delta x}$$

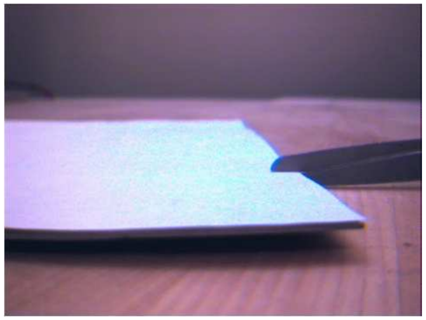
Can we always guarantee when a task is achieved/achievable?

Task ambiguity

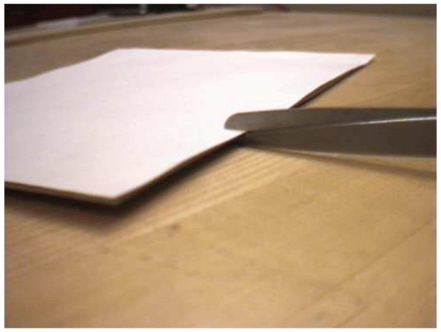




- Will the scissors cut the paper in the middle?

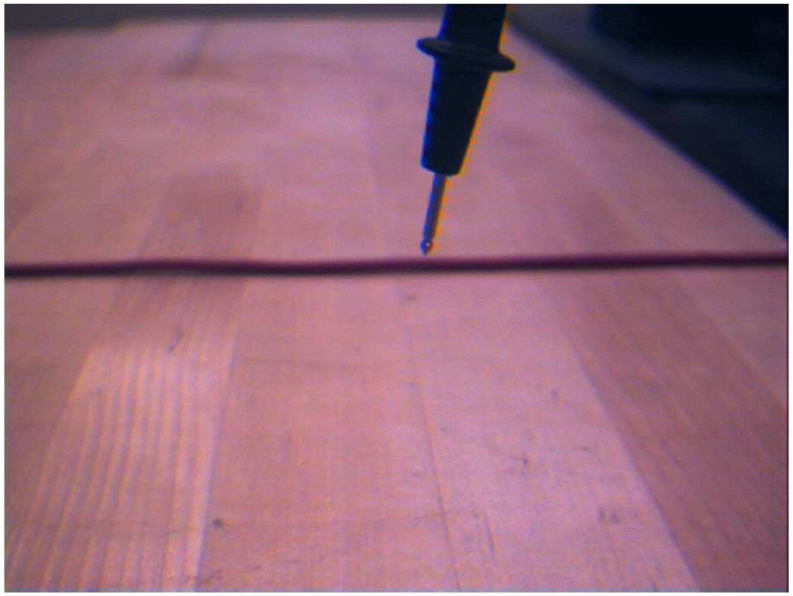


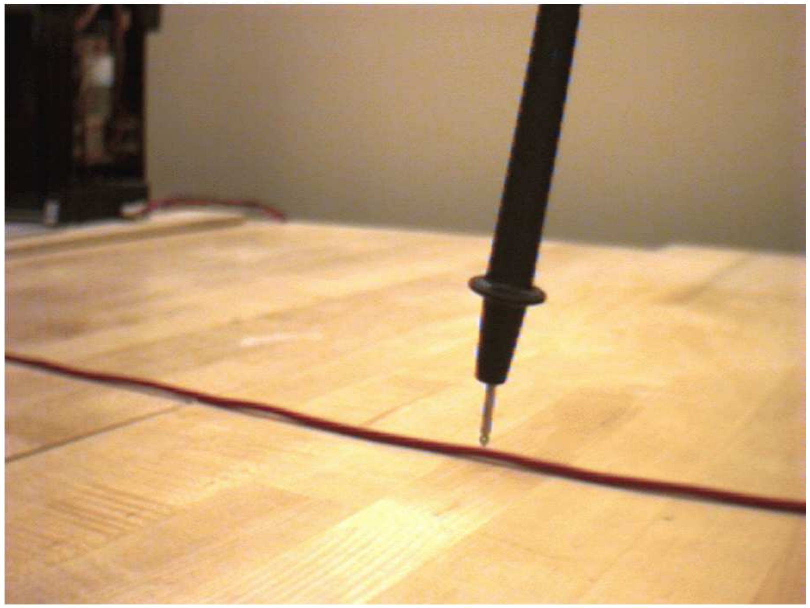
Task ambiguity



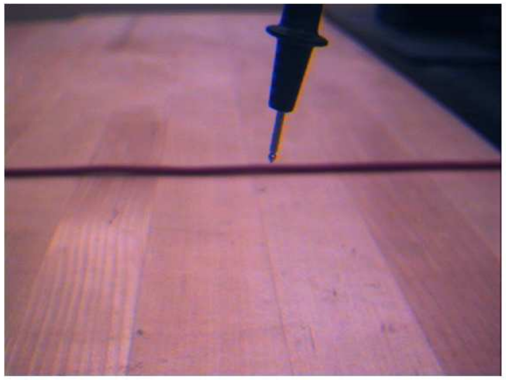
- Will the scissors cut the paper in the middle? **NO!**

Task Ambiguity

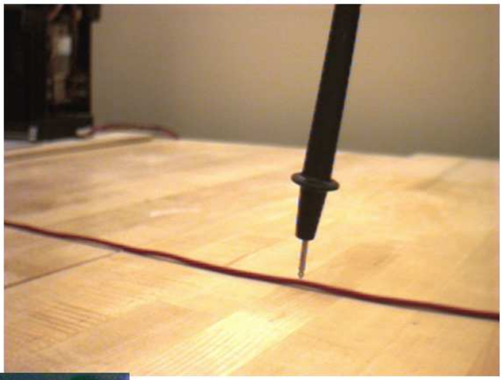


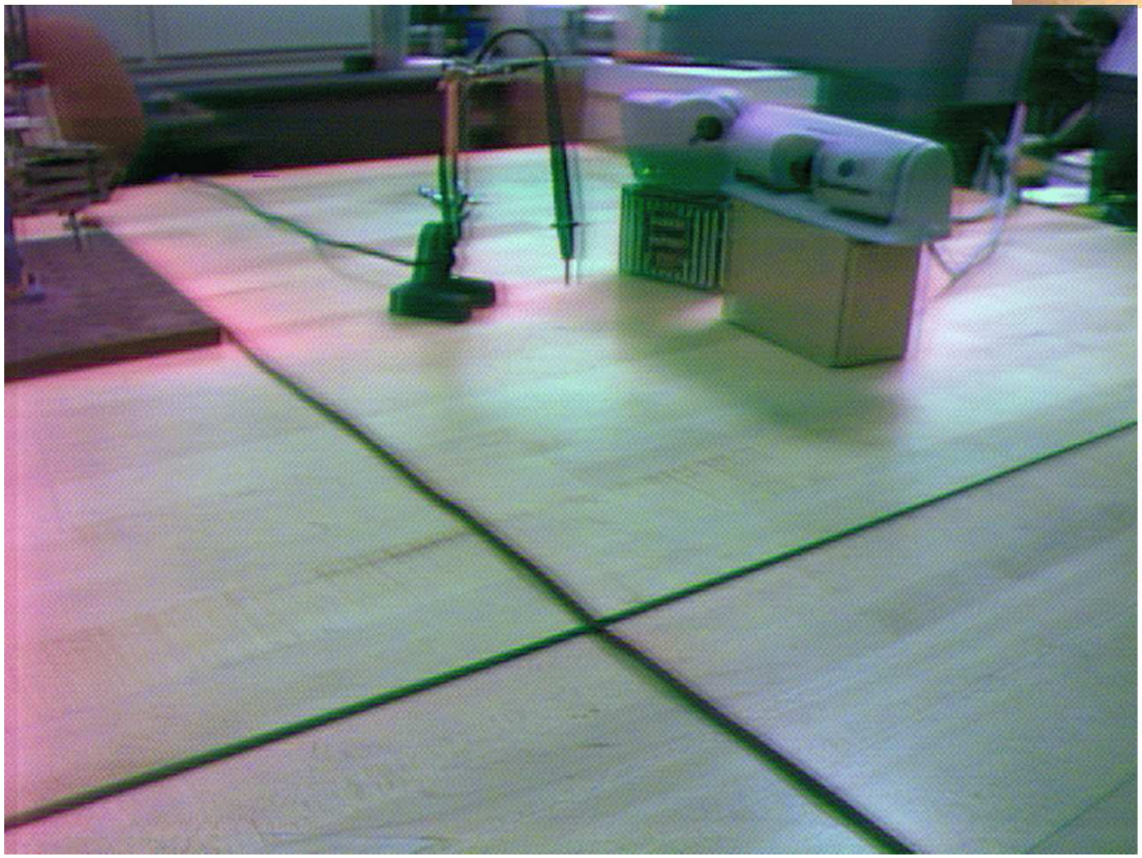


- Is the probe contacting the wire?



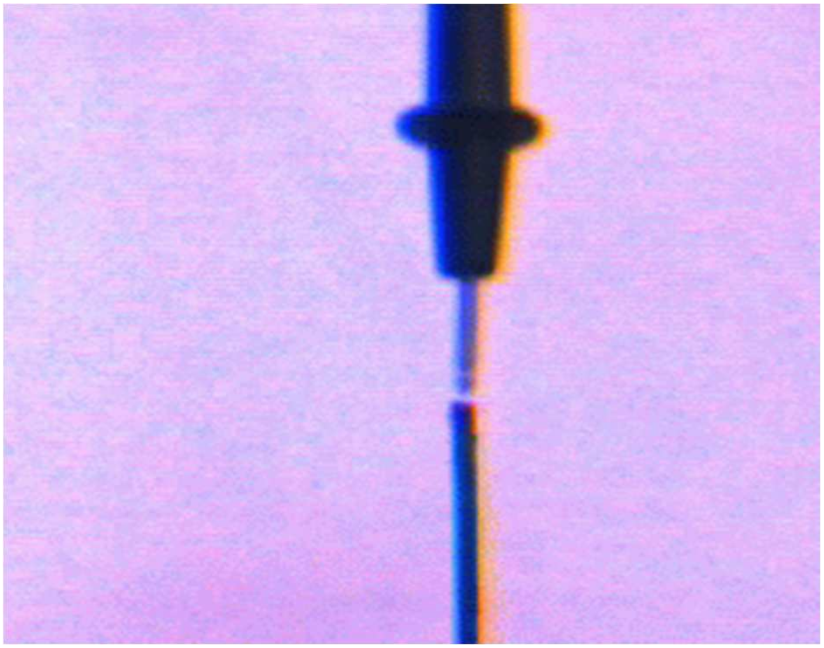
Task Ambiguity

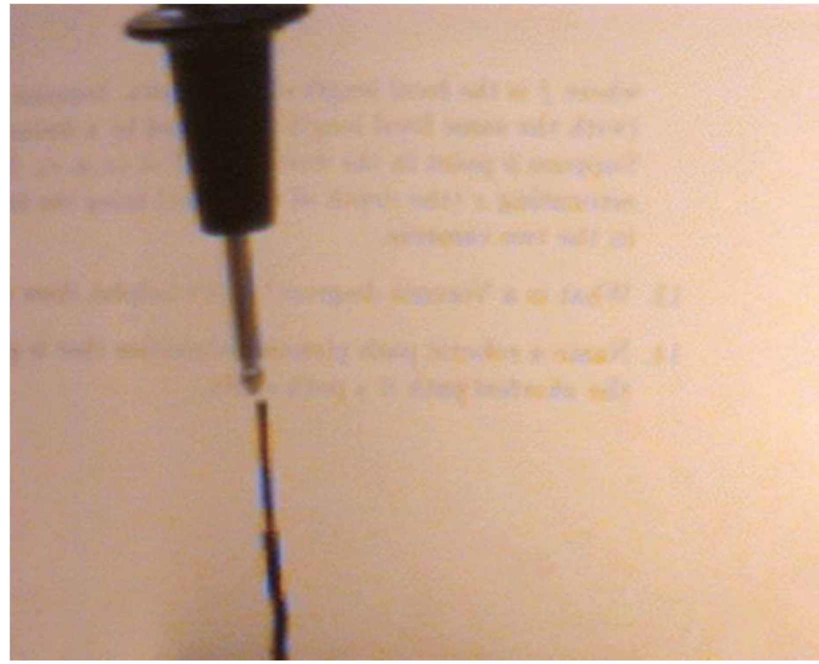




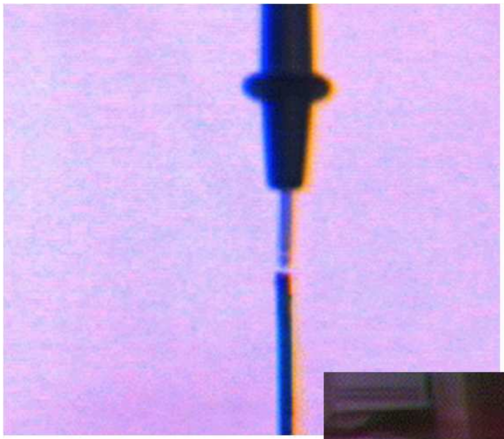
- Is the probe contacting the wire? **NO!**

Task Ambiguity

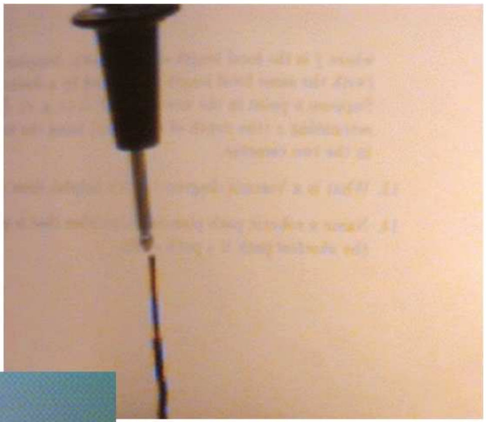


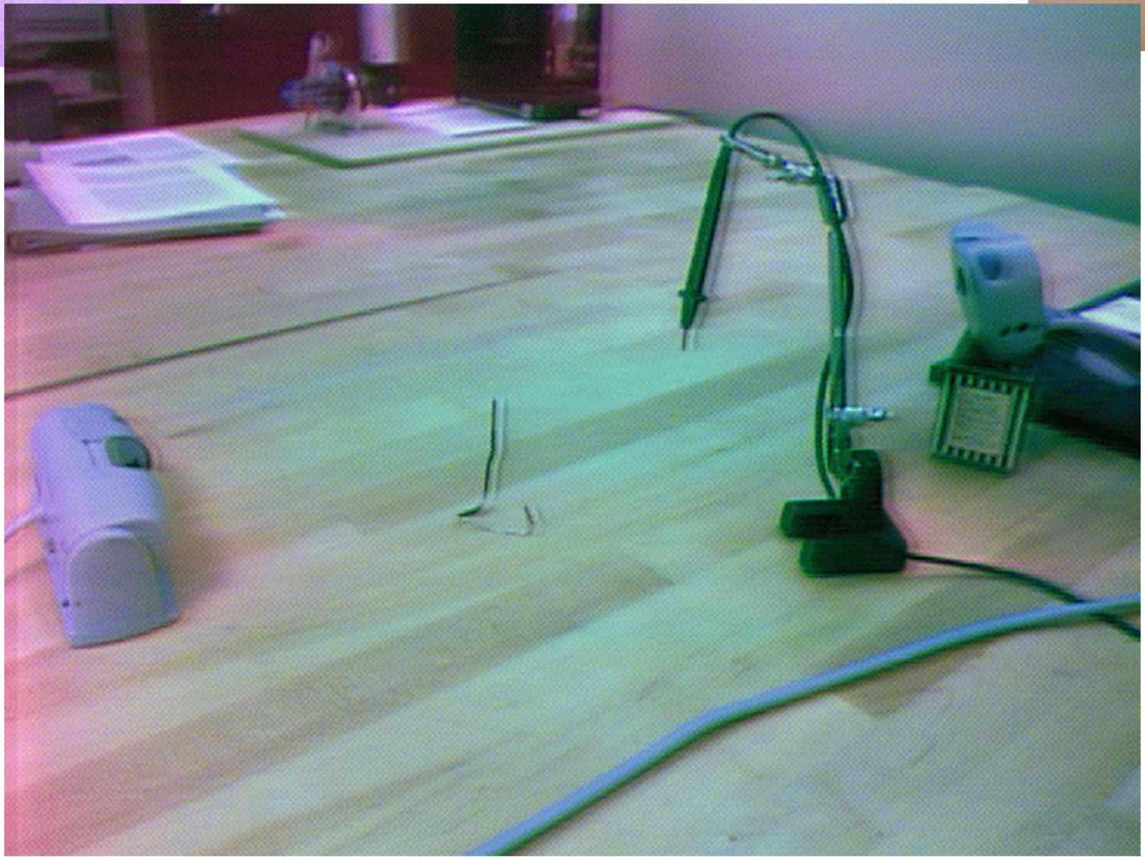


- Is the probe contacting the wire?



Task Ambiguity





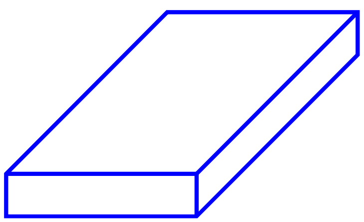
- Is the probe contacting the wire? **NO!**

Task decidability and invariance

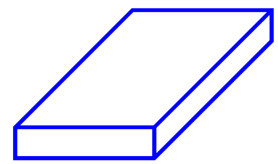
A task $T(f)=0$ is invariant under a group G_x of transformations, iff

$$\forall\, f \in V^n,\ g \in G_x \text{ with } g(f) \in V^n \qquad T(f)=0 \Leftrightarrow T(g(f))=0$$

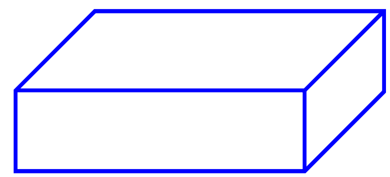
If $T(f) = 0$ here,



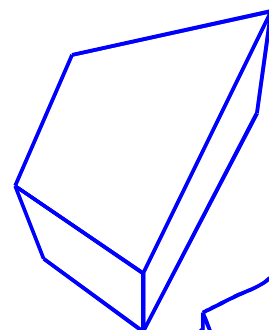
$T(f)$ must be 0 here, if T is G_{sim} invariant



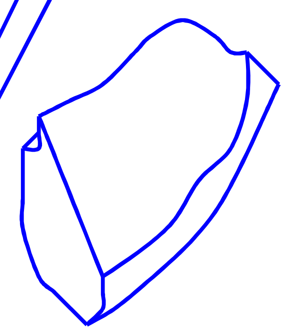
$T(f)$ must be 0 here, if T is G_{aff} invariant



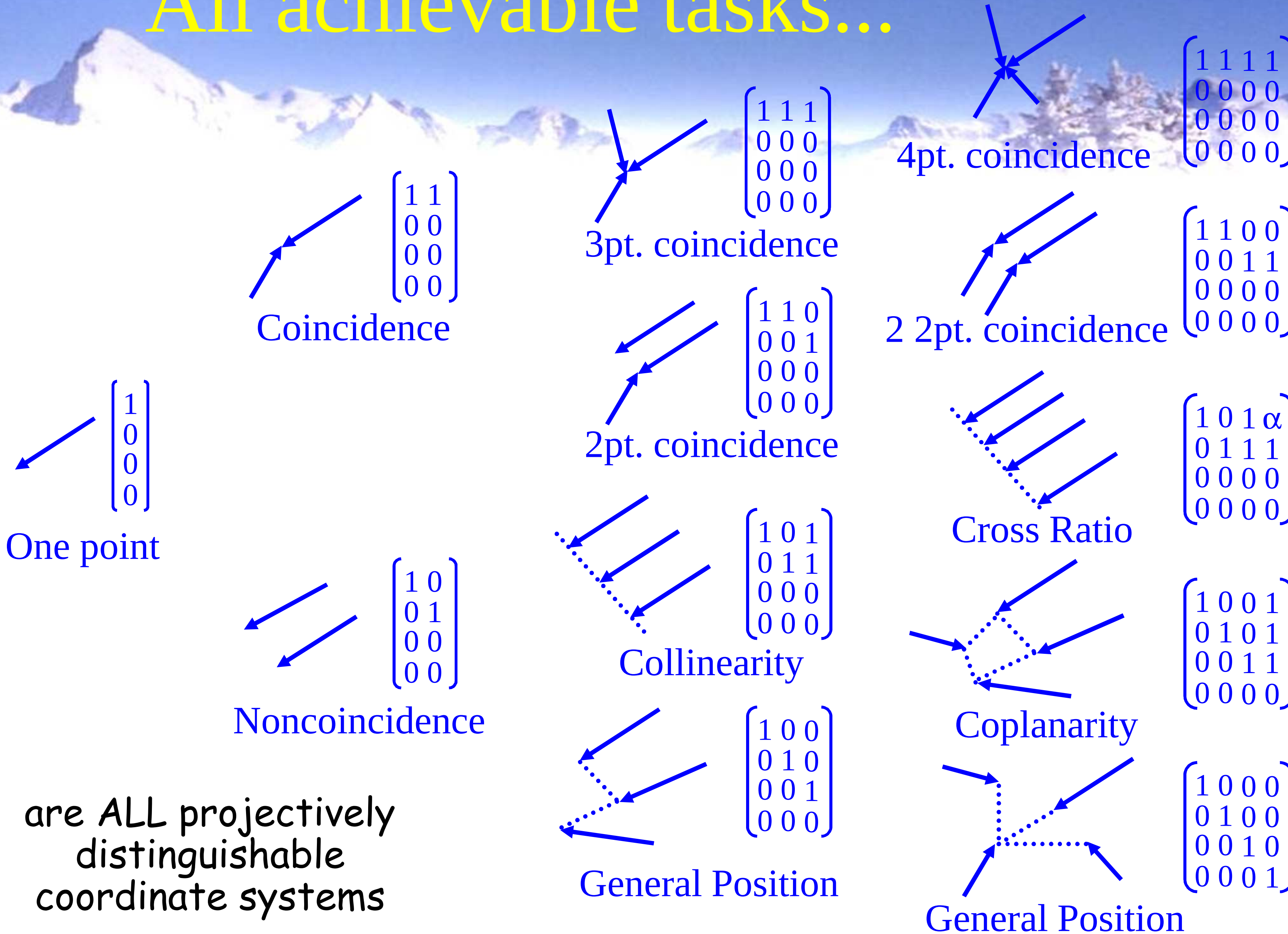
$T(f)$ must be 0 here, if T is G_{proj} invariant



$T(f)$ must be 0 here, if T is G_{inj} invariant

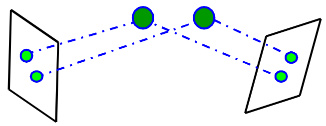


All achievable tasks...



are ALL projectively distinguishable coordinate systems

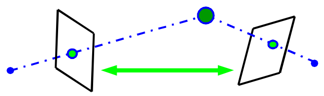
Composing possible tasks



Injective C_{inj}

T_{pp} point-coincidence

Task primitives



Projective C_{wk}

T_{pp}

T_{col} collinearity

T_{copl} coplanarity

$T_{cr\alpha}$ cross ratios (α)

AND

$$T_1 \wedge T_2 = \begin{pmatrix} T_1 \\ T_2 \end{pmatrix}$$

OR

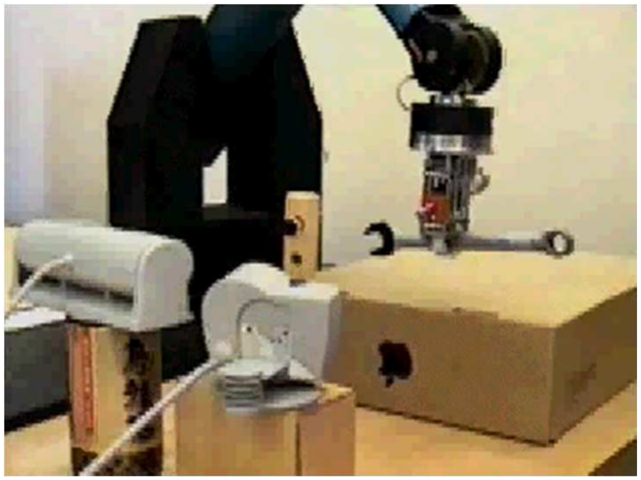
$$T_1 \vee T_2 = T_1 \cdot T_2$$

NOT

$$\neg T = \begin{cases} 0 & \text{if } T \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

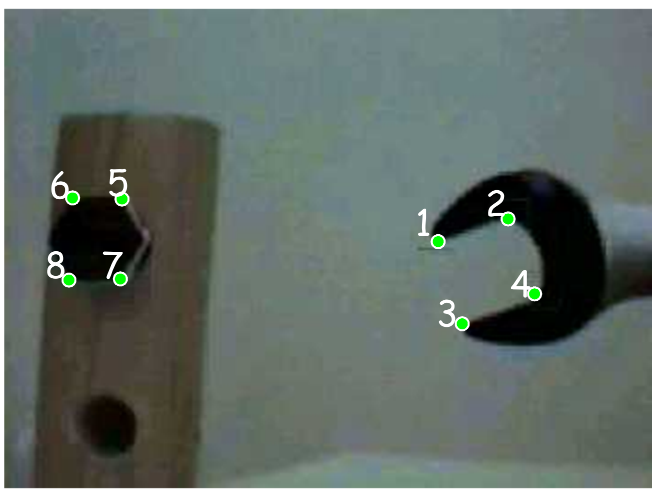
Task operators

Result: Task toolkit



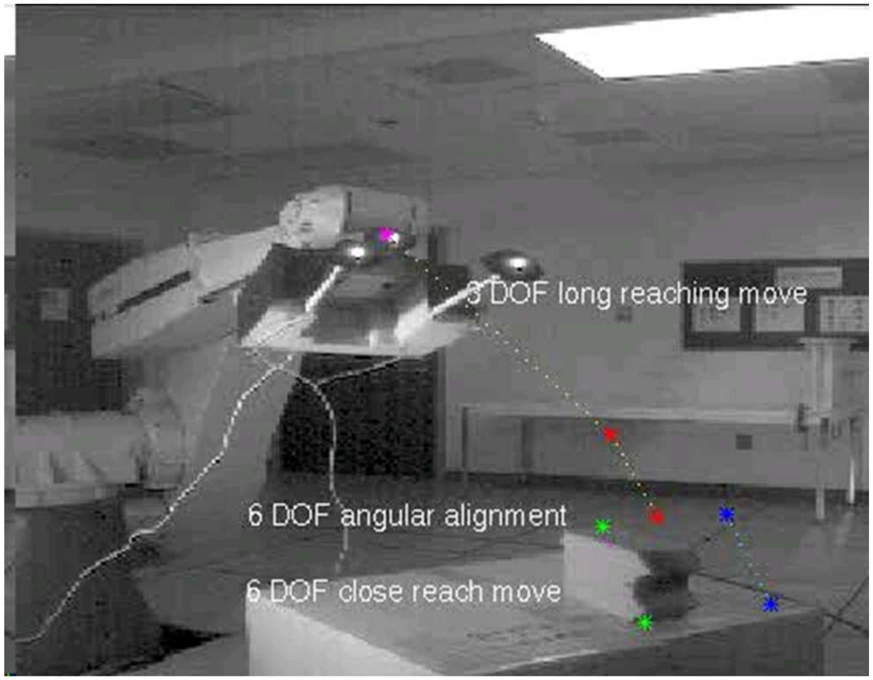
wrench: view 1

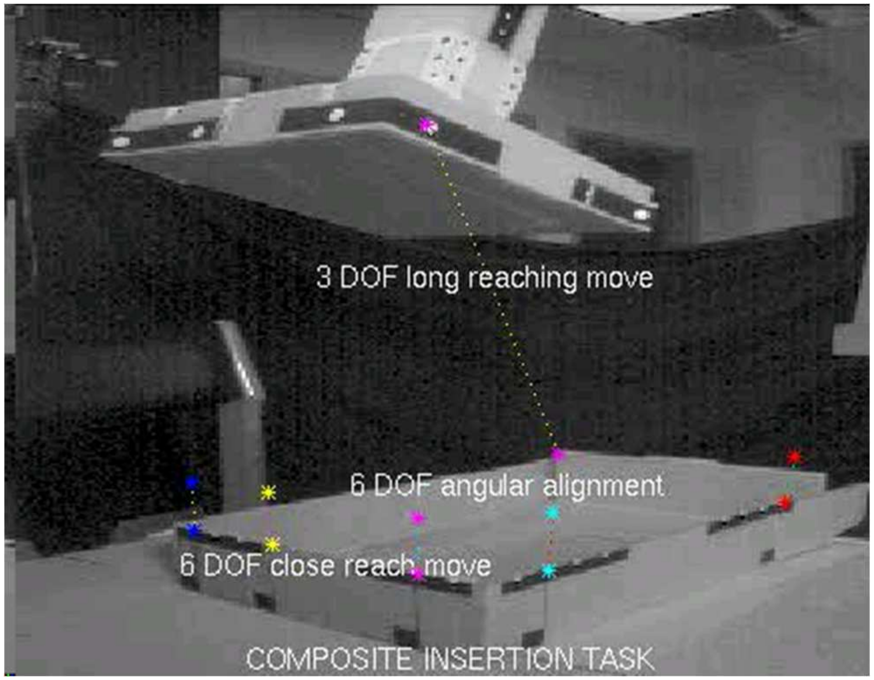
wrench: view 2



$$T_{wrench}(x_{1..8}) = \begin{array}{l} T_{pp}(x_1,x_5) \wedge \\ T_{pp}(x_3,x_7) \wedge \\ T_{col}(x_4,x_7,x_8) \wedge \\ T_{col}(x_2,x_5,x_6) \end{array}$$

Example:
Pick and place type of movement





3. Alignment???

- To match transport final to fine manipulation initial conditions

More primitives

4. Guarded move
   – Move along some direction until an external contraint (e.g. contact) is satisfied.

5. Open loop movements:
   - When object is obscured
   - Or ballistic fast movements
   - Note can be done based on previously estimated Jacobians

Solving the puzzle…

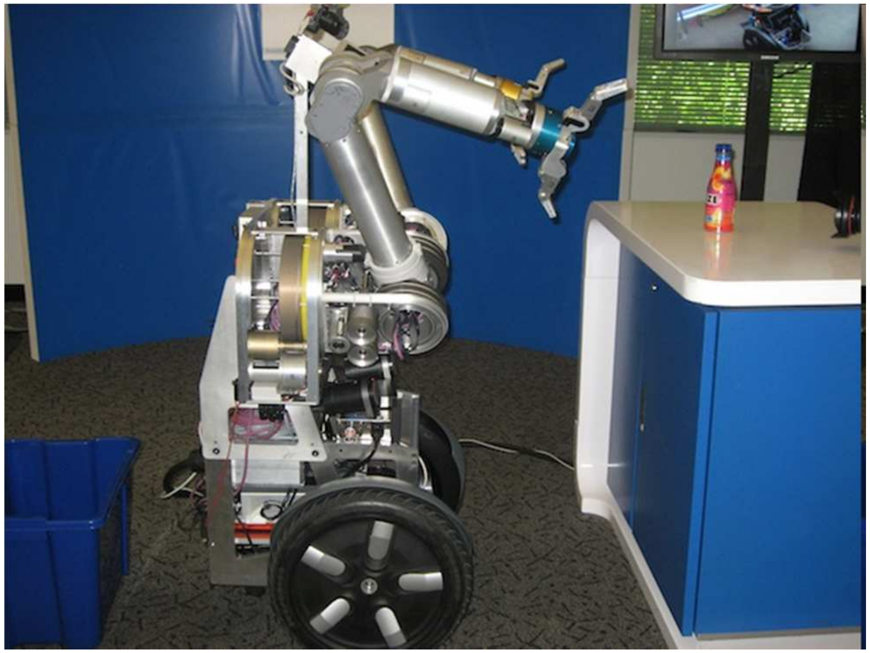
Conclusions

- Visual servoing: The process of minimizing a visually-specified task by using visual feedback for motion control of a robot.

- Is it *difficult*? Yes.
  - Controlling 6D pose of the end-effector from 2D image features.
  - Nonlinear projection, degenerate features, etc.
- Is it important? Of course.
  - Vision is a versatile sensor.
  - Many applications: industrial, health, service, space, humanoids, etc.

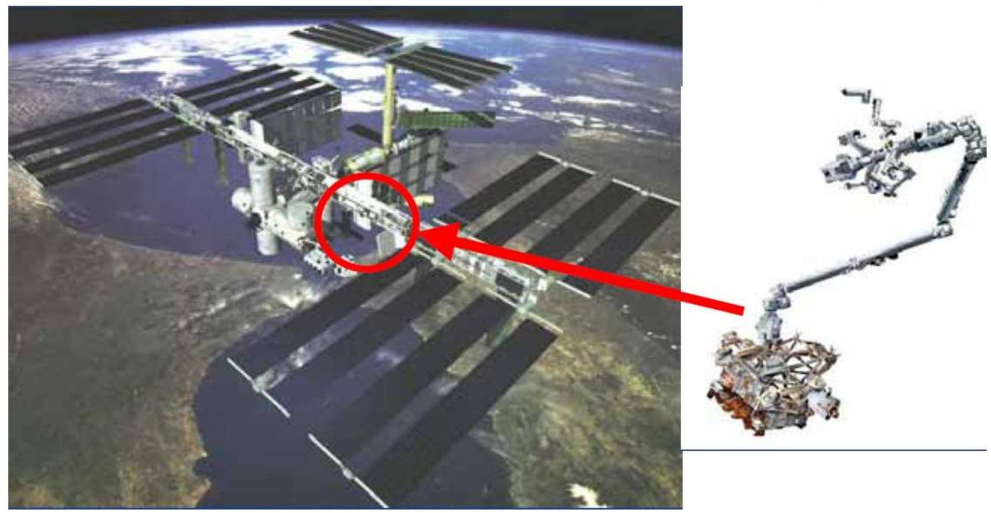
What environments and tasks are of interest?

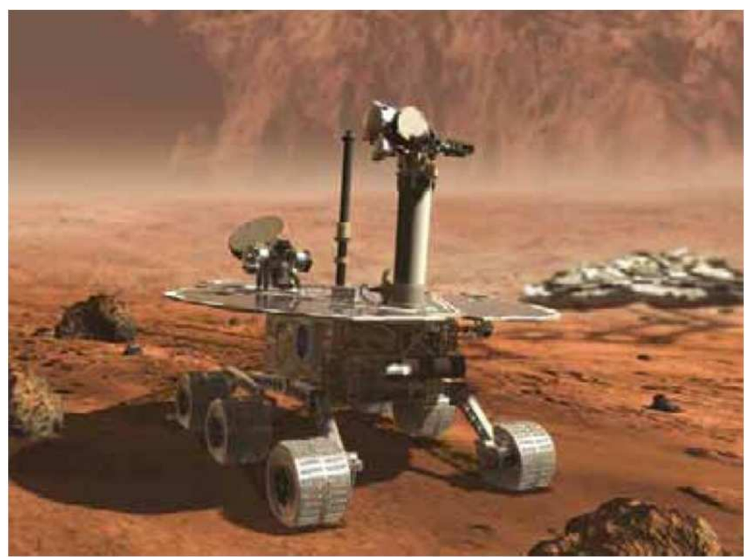
- Not the previous examples.
  - Could be solved “structured”
- Want to work in everyday unstructured environments



Use in space applications

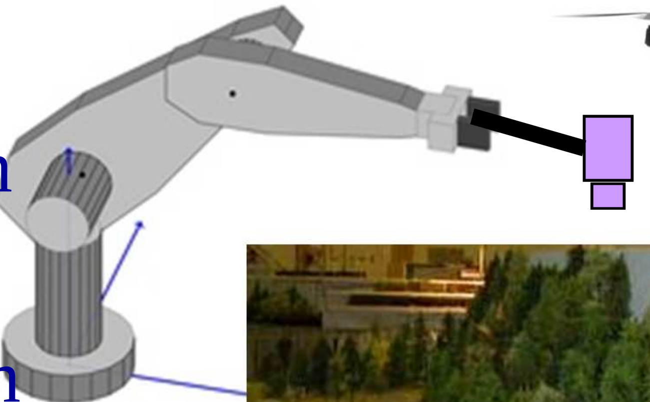
- Space station
- On-orbit service
- Planetary Exploration



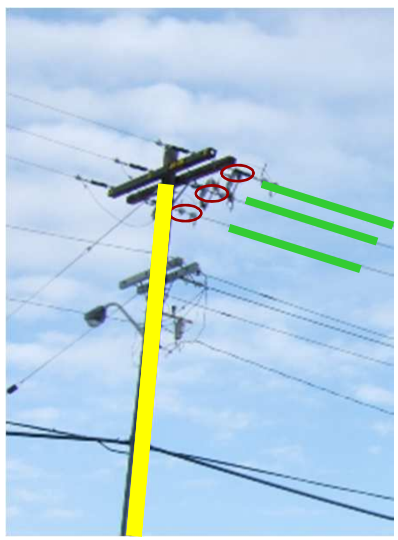


Power line inspection

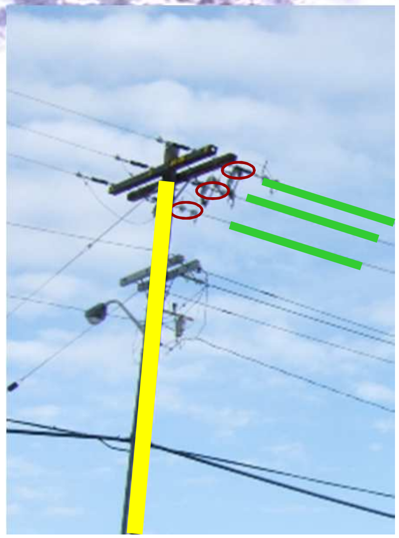
- A electric UAV carrying camera flown over the model
- Simulation: Robot arm holds camera



Model landscape: Edmonton railroad society



Use in Power Line Inspection



Some References

- M. Spong, S. Hutchinson, M. Vidyasagar. Robot Modeling and Control. Chapter 12: Vision-Based Control. John Wiley & Sons: USA, 2006. (Text)
- S. Hutchinson, G. Hager, P. Corke, "A tutorial on visual servo control," IEEE Trans. on Robotics and Automation, October 1996, vol. 12, no 5, p. 651-670. (Tutorial)
- F. Chaumette, S. Hutchinson, "Visual Servo Control, Part I: Basic Approaches," and "Part II: Advanced Approaches," IEEE Robotics and Automation Magazine 13(4):82-90, December 2006 and 14(1):109-118, March 2007. (Tutorial)
- B. Espiau, F. Chaumette, P. Rives, "A new approach to visual servoing in robotics," IEEE Trans. on Robotics and Automation, June 1992, vol. 8, no. 6, p. 313-326. (Image-based)
- W. J. Wilson, C. C. Williams Hulls, G. S. Bell, "Relative End-Effector Control Using Cartesian Position Based Visual Servoing", IEEE Transactions on Robotics and Automation, Vol. 12, No. 5, October 1996, pp.684-696. (Position-based)
- E. Malis, F. Chaumette, S. Boudet, "2 1/2 D visual servoing," IEEE Trans. on Robotics and Automation, April 1999, vol. 15, no. 2, p. 238-250. (Hybrid)
- M. Jägersand, O. Fuentes, R. Nelson, "Experimental Evaluation of Uncalibrated Visual Servoing for Precision Manipulation," Proc. IEEE Intl. Conference on Robotics and Automation (ICRA), pp. 2874-2880, 20-25 Apr. 1997. (Uncalibrated, model-free)
- F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," The Confluence of Vision and Control, LNCS Series, No 237, Springer-Verlag, 1998, p. 66-78.