Perception & Sensing in Robotic Mobility and Manipulation

Gregory D. Hager
Laboratory for Computation, Sensing, and Control
Department of Computer Science
Johns Hopkins University

The Role of Perception in RMM

- Where am I relative to the world?
  - sensors: vision, stereo, range sensors, acoustics
  - problems: scene modeling/classification/recognition
  - integration: localization/mapping algorithms (e.g. SLAM)
- What is around me?
  - sensors: vision, stereo, range sensors, acoustics, sounds, smell
  - problems: object recognition, structure from x, qualitative modeling
  - integration: collision avoidance/navigation, learning

The Role of Perception in RMM

- How can I safely interact with environment (including people!)?
  - sensors: vision, range, haptics (force+tactile)
  - problems: structure/range estimation, modeling, tracking, materials, size, weight, inference
  - integration: navigation, manipulation, control, learning
- How can I solve “new” problems (generalization)?
  - sensors: vision, range, haptics, undefined new sensor
  - problems: categorization by function/shape/context/??
  - integrate: inference, navigation, manipulation, control, learning

Topics Today

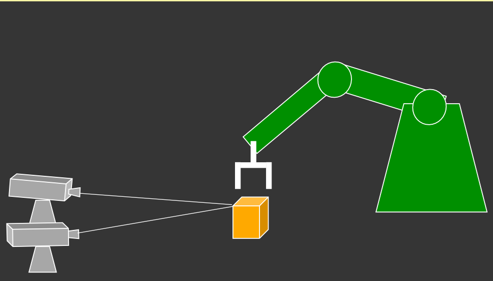
Techniques

- Computational Stereo
- Feature detection and matching
- Motion tracking and visual feedback

Applications in Robotics:

- Obstacle detection, environment interaction
- Mapping, registration, localization, recognition
- Manipulation

What is Computational Stereo?



Viewing the same physical point from two different viewpoints allows depth from triangulation

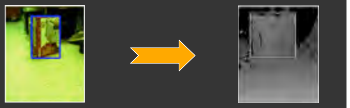
Computational Stereo

- Much of geometric vision is based on information from 2 (or more) camera locations
  - hard to recover 3D information from a single 2D image without extra knowledge
  - motion and stereo (multiple cameras) are both common in the world
- Stereo vision is ubiquitous in nature
  - (oddly, nearly 10% of people are stereo blind)
- Stereo involves the following *three problems*:

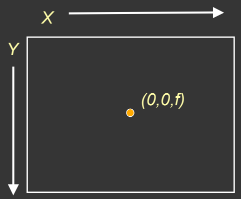
1. calibration
2. matching (correspondence problem)
3. reconstruction (reconstruction problem)

Binocular Stereo System: Geometry

- **GOAL:** Passive 2-camera system using triangulation to generate a depth map of a world scene.



- **Depth map:** z=f(x,y) where x,y are coordinates one of the image planes and z is the height above the respective image plane.
  - Note that for stereo systems which differ only by an offset in x, the v coordinates (projection of y) is the same in both images!
  - Note we must convert from image (pixel) coordinates to external coordinates -- **requires calibration**



4 intrinsic parameters convert from pixel to metric values
s_x s_y c_x c_y

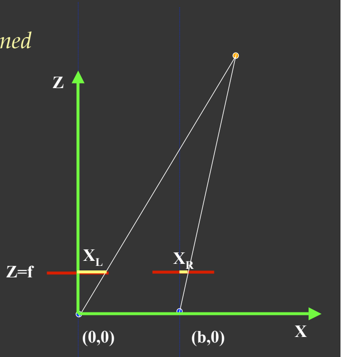
Non-verged Binocular Stereo System

Assume: image are scan-line aligned

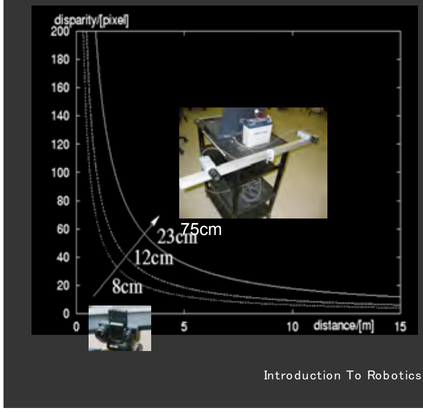
From perspective projection:
$x_L = s_x X/Z$
$x_R = s_x (X - b)/Z$
$y_L = y_R = s_y Y/Z$

Define Disparity:
$D = (x_L - x_R)$

$$Z = \frac{b\, s_x}{D}$$



Stereo-System Accuracy

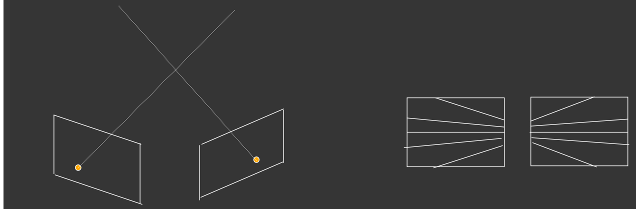


$$Z = \frac{b\, s_x}{D}$$

To increase resolution:

- Increase of the baseline (B) - size of the system
- Increase of the focal length (f) - field of view
- Decrease of the pixel-size ($1/s_x$) - resolution of the camera

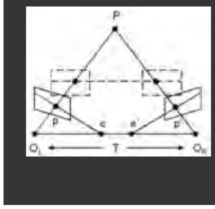
Two-Camera Geometry

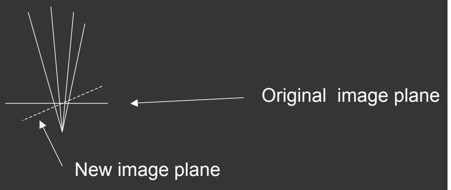


It is not hard to show that when we rotate the cameras inward, corresponding points no longer lie on a scan line

How to Change Epipolar Geometry

Image rectification is the computation of an image as seen by a rotated camera

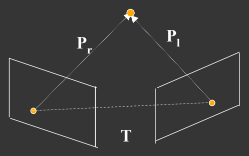




Fundamental Matrix Derivation

Note that E is invariant to the scale of the points, therefore we also have

$$p_r^t E\, p_l = 0$$



$$P_r = R(P_l - T)$$

where p denotes the (metric) image projection of P

Now if K denotes the internal calibration, converting from metric to pixel coordinates, we have further that

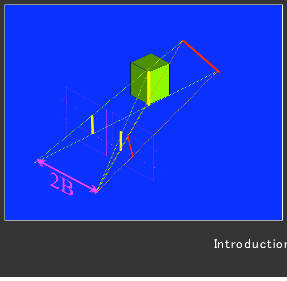
$$r_r^t K^{-t} E\, K^{-1} r_l = r_r^t F\, r_l = 0$$

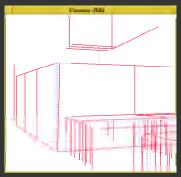
where r denotes the *pixel* coordinates of p. F is called the *fundamental matrix*

Stereo-Based Reconstruction

Correspondence Problem:

How to find corresponding areas of two camera images (points, line segments, curves, regions)

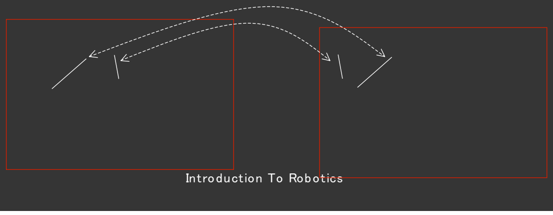




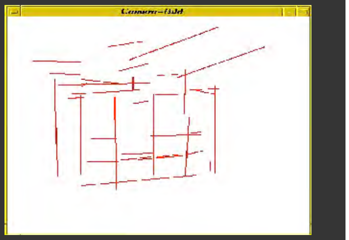
MATCHING AND CORRESPONDENCE

- Two major approaches
  - feature-based
  - region based

In feature-based matching, the idea is to pick a feature type (e.g. edges), define a matching criteria (e.g. orientation and contrast sign), and then look for matches within a disparity range



Results - Reconstruction

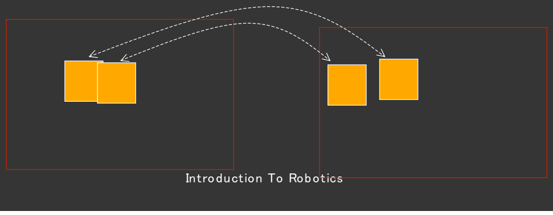


MATCHING AND CORRESPONDENCE

- Two major approaches
  - feature-based
  - region based

In region-based matching, the idea is to pick a region in the image and attempt to find the matching region in the second image by maximizing the some measure:

1. normalized SSD
2. SAD
3. normalized cross-correlation

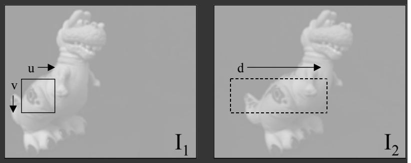


Match Metric Summary

MATCH METRIC	DEFINITION
Normalized Cross-Correlation (NCC)	$\dfrac{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)\cdot\left(I_2(u+d,v)-\bar{I}_2\right)}{\sqrt{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)^2\cdot\sum_{u,v}\left(I_2(u+d,v)-\bar{I}_2\right)^2}}$
Sum of Squared Differences (SSD)	$\sum_{u,v}\left(I_1(u,v)-I_2(u+d,v)\right)^2$
Normalized SSD	$\sum_{u,v}\left(\dfrac{\left(I_1(u,v)-\bar{I}_1\right)}{\sqrt{\sum_{u,v}\left(I_1(u,v)-\bar{I}_1\right)^2}}-\dfrac{\left(I_2(u+d,v)-\bar{I}_2\right)}{\sqrt{\sum_{u,v}\left(I_2(u+d,v)-\bar{I}_2\right)^2}}\right)^2$
Sum of Absolute Differences (SAD) Zero Mean SAD	$\sum \left\|I_1(u,v)-I_2(u+d,v)\right\|$ $\sum_{u,v}\left\|(I_1(u,v)-\bar{I}_1)-(I_2(u+d,v)-\bar{I}_2)\right\|$
Rank	$I'_k(u,v)=\sum_{m,n}I_k(m,n)<I_k(u,v)$ $\sum_{u,v}\left(I'_1(u,v)-I'_2(u+d,v)\right)$
Census	$I'_k(u,v)=BITSTRING_{m,n}\left(I_k(m,n)<I_k(u,v)\right)$ $\sum_{u,v}HAMMING\left(I'_1(u,v),I'_2(u+d,v)\right)$

Remember, these two are actually the same

Correspondence Search Algorithm



```
For  i = 1:nrows
  for j=1:ncols
          best(i,j) = -1
          for k = mindisparity:maxdisparity
             c = ComputeMatchMetric($I_1$(i,j),$I_2$(i,j+k),winsize)
             if (c > best(i,j))
                     best(i,j) = c
                     disparities(i,j) = k
           end
         end
    end
end
```

O(nrows * ncols * disparities * winx * winy)

Correspondence Search Algorithm V2

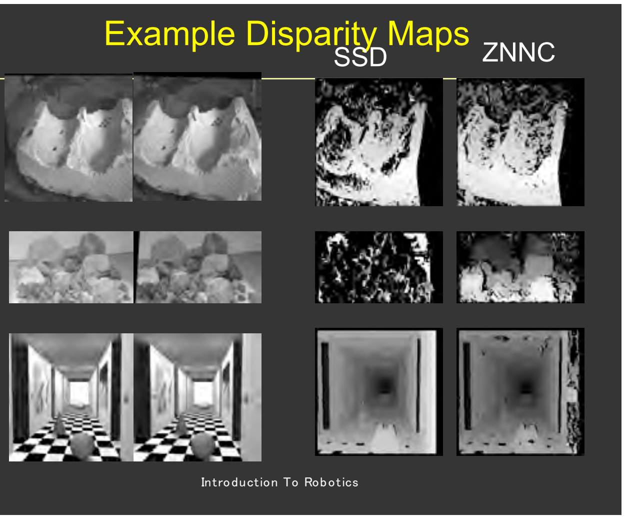
```
best = -ones(size(im))
disp = zeros(size(im))
for k = mindisparity:maxdisparity
        prod = $I_1$(:,overlap) .* $I_2$(:,k+overlap)
        CC = conv2(prod,fspecial(‘average’,winsize))
        better = CC > best;
        disp = better .* k + (1-better).*disp;
        best = better .*CC + (1-better).*best;
end
```

Typically saves O(winx*winy) operations for most any match metric

An Additional Twist

- Note that searching from left to right *is not the same* as searching from right to left.
- As a result, we can obtain a somewhat independent disparity map by flipping the images around.
- The results should be the same map up to sign.
- LRCheck: $disp_{lr}(i,j) = - disp_{rl}(i,j+disp_{lr}(i,j))$

Example Disparity Maps



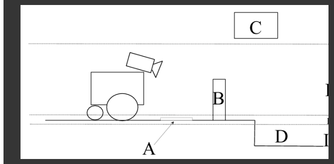
Real-Time Stereo

REAL-TIME STEREO SYSTEM	IMAGE SIZE	FRAME RATE	RANGE BINS	METHOD	PROCESSOR	CAMERAS
INRIA 1993	256x256	3.6 fps	32	Normalized Correlation	PeRLe-1	3
CMU iWarp 1993	256x240	15 fps	16	SSAD	64 Processor iWarp Computer	3
Teleos 1995	320x240	0.5 fps	32	Sign Correlation	Pentium 166 MHz	2
JPL 1995	256x240	1.7 fps	32	SSD	Datacube & 68040	2
CMU Stereo Machine 1995	256x240	30 fps	30	SSAD	Custom HW & C40 DSP Array	6
Point Grey Triclops 1997	320x240	6 fps	32	SAD	Pentium II 450 MHz	3
SRI SVS 1997	320x240	12 fps	32	SAD	Pentium II 233 MHz	2
SRI SVM II 1997	320x240	30+ fps	32	SAD	TMS320C60x 200MHz DSP	2
Interval PARTS Engine 1997	320x240	42 fps	24	Census Matching	Custom FPGA	2
CSIRO 1997	256x256	30 fps	32	Census Matching	Custom FPGA	2
SAZAN 1999	320x240	20 fps	25	SSAD	FPGA & Convolvers	9
Point Grey Triclops 2001	320x240	20 fps 13 fps	32	SAD	Pentium IV 1.4 GHz	2 3
SRI SVS 2001	320x240	30 fps	32	SAD	Pentium III 700 MHZ	2

Applications of Real-Time Stereo

- Mobile robotics
  - Detect the structure of ground; detect obstacles; convoying
- Graphics/video
  - Detect foreground objects and matte in other objects (super-matrix effect)
- Surveillance
  - Detect and classify vehicles on a street or in a parking garage
- Medical
  - Measurement (e.g. sizing tumors)
  - Visualization (e.g. register with pre-operative CT)

Stereo Example: Obstacle Detection

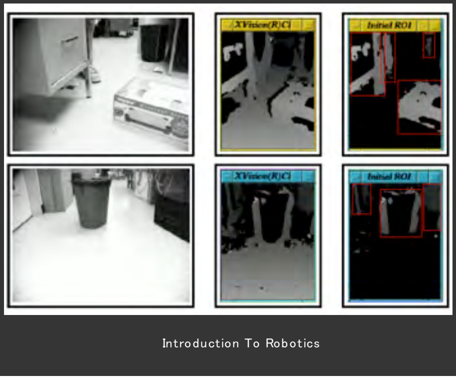


Problem to solve:

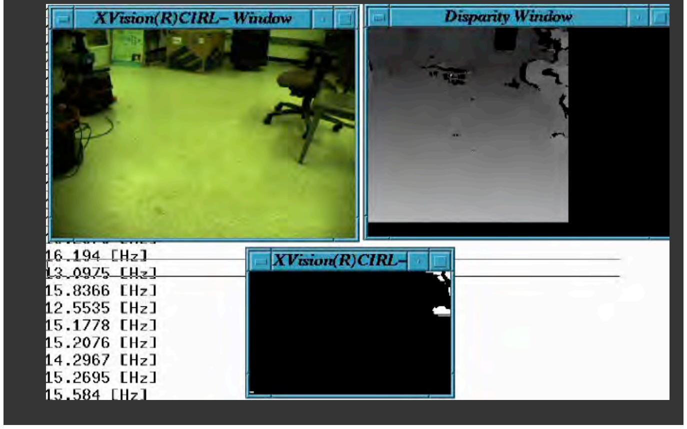
Distinguish between relevant obstacles (B,D) and irrelevant (A,C) obstacles

Obstacle Detection (cont'd)

Observation: Removing the ground plane immediately exposes obstacles



Applications of Real-Time Stereo



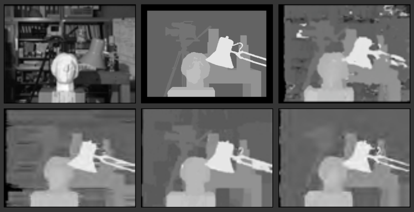
```
16.194 [Hz]
13.0975 [Hz]
15.8366 [Hz]
12.5535 [Hz]
15.1778 [Hz]
15.2076 [Hz]
14.2967 [Hz]
15.2695 [Hz]
15.584 [Hz]
```

Other Problems:

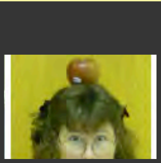
- Photometric issues:
  - specularities
  - strongly non-Lambertian BRDF's
- Surface structure
  - lack of texture
  - repeating texture within horopter bracket
- Geometric ambiguities
  - as surfaces turn away, difficult to get accurate reconstruction (affine approximate can help)
  - at the occluding contour, likelihood of good match but incorrect reconstruction

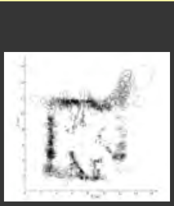
Local vs. Global Matching

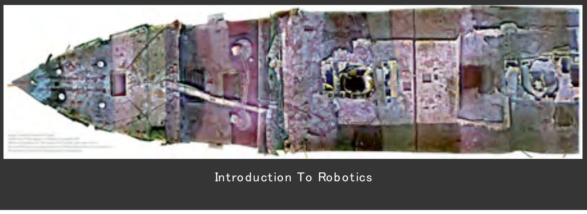
Comparative results on images from the University of Tsukuba, provided by Scharstein and Szeliski [69]. Left to right: left stereo image, ground truth, Muhlmann et al.'s area correlation algorithm [57], dynamic programming (similar to Intille and Bobick [36]), Roy and Cox's maximum flow [65] and Komolgorov and Zabih's graph cuts [45].



Mapping, Localization, Recognition



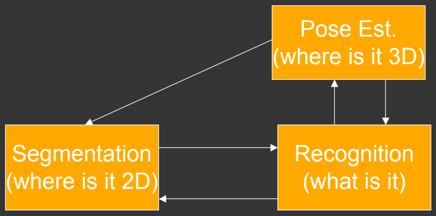




Object Recognition: The Problem

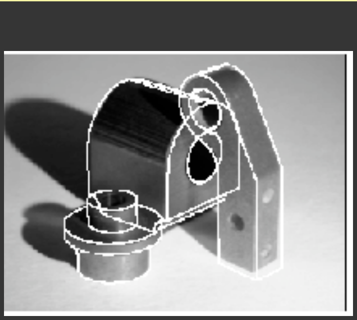
Given: A database D of "known" objects and an image I:

1. Determine which (if any) objects in D appear in I
2. Determine the pose (rotation and translation) of the object



The object recognition conundrum

Recognition From Geometry?

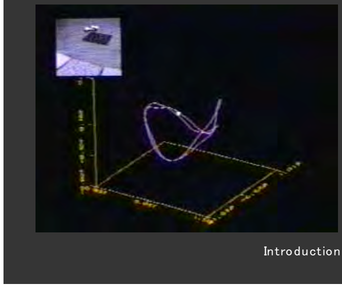


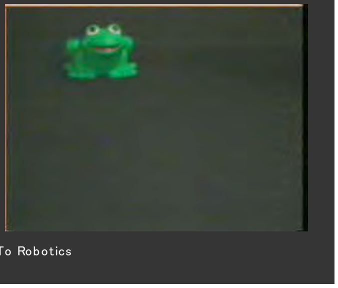
Given a database of objects and an image determine what, if any of the objects are present in the image.

Recognition From Appearance?

- Columbia SLAM system:
  - can handle databases of 100's of objects
  - single change in point of view
  - uniform lighting conditions

Courtesy Shree Nayar, Columbia U.



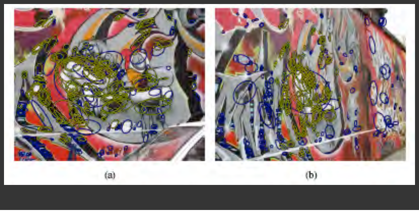


Current Best Solution

- Generally view based
- Uses local features and "local" invariance (global is too weak)
- Uses *lots* of features and some sort of voting
- Also recent attempts to perform "categorical" object recognition using similar techniques
- Example: recent papers by Schmid, Lowe, Ponce, Hebert, Perona ...
- Here, we discus SIFT features (Lowe 1999)

Feature Desiderata

- Features should be distinctive
- Features should be easily detected under changes in pose, lighting, etc.
- There should be many features per object

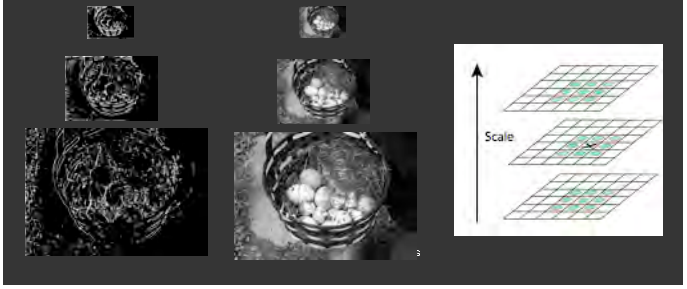


Steps in SIFT Feature Selection

- Scale-space peak selection
- Keypoint localization
  - includes rejection due to poor localization
  - also perform cornerness check using eigenvalues; reject those with eigenvalue ratio greater than 10
- Orientation Assignment
  - dominant orientation plus any within 80% of dominant
- Build keypoint descriptor
- Normal images yield approx. 2000 stable features
  - small objects in cluttered backgrounds require 3-6 features

Peak Detection

- Find all max and min is LoG images in both space and scale
  - 8 spatial neighbors; 9 scale neighbors
  - orientation based on maximum of weighted histogram



Keypoint Descriptor

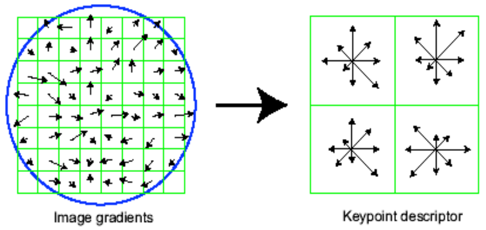


Figure 7: A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point, as shown on the left. These are weighted by a Gaussian window, indicated by the overlayed circle. These samples are then accumulated into orientation histograms summarizing the contents over larger regions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. To reduce clutter, this figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas most experiments in this paper use 4x4 descriptors computed from a 16x16 sample array.

Example

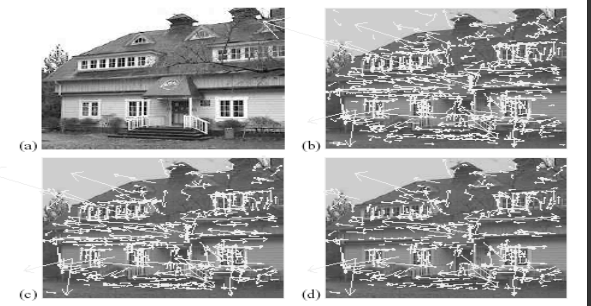


Figure 5: This figure shows the stages of keypoint selection. (a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principle curvatures.

PDF of Matching

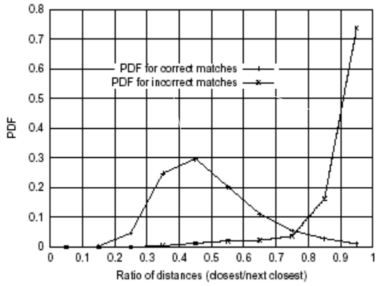


Figure 11: The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbor to the distance of the second closest. Using a database of 40,000 keypoints, the solid line shows the PDF of this ratio for correct matches, while the dotted line is for matches that were incorrect.

Feature Matching

- Uses a Hough transform (voting technique)
  - parameters are position, orientation and scale for each training view
  - features are matched to closest Euclidean distance neighbor in database; each database feature indexed to object and view as well as location, orientation and scale
  - features are linked to adjacent model views; these links are also followed and accumulated
  - implemented using a hash table

Results

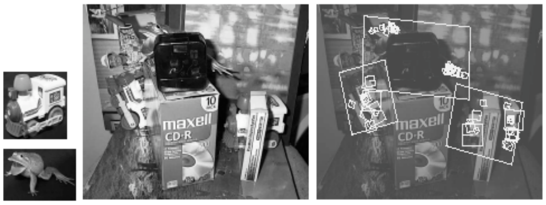
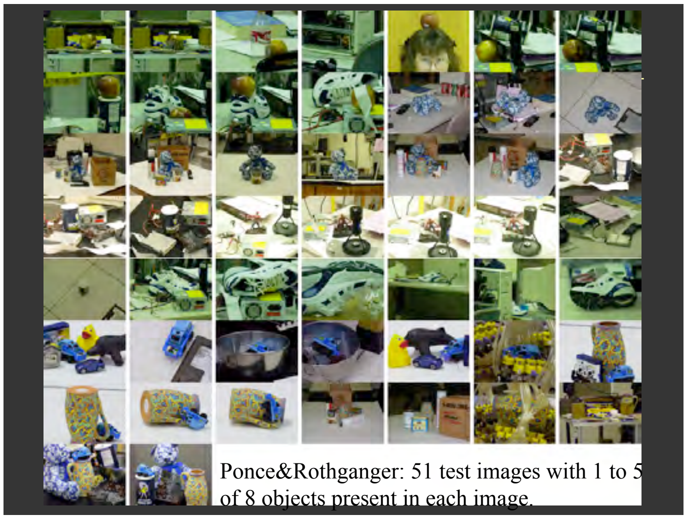
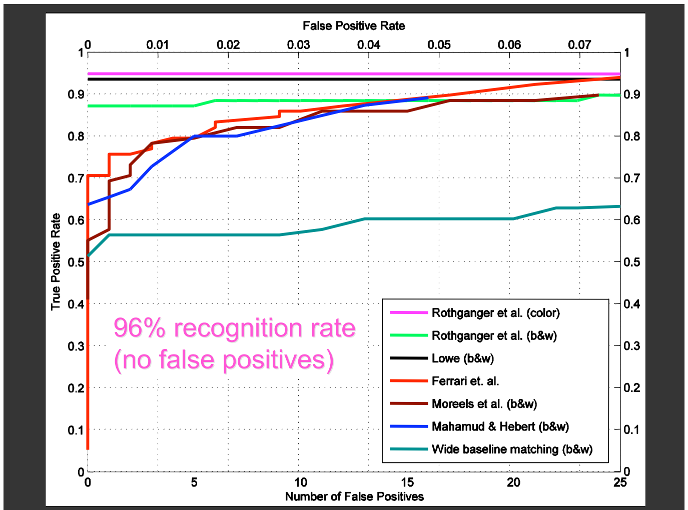


Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right overlaid on a reduced contrast version of the image. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affine transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.



Ponce&Rothganger: 51 test images with 1 to 5 of 8 objects present in each image.



Results

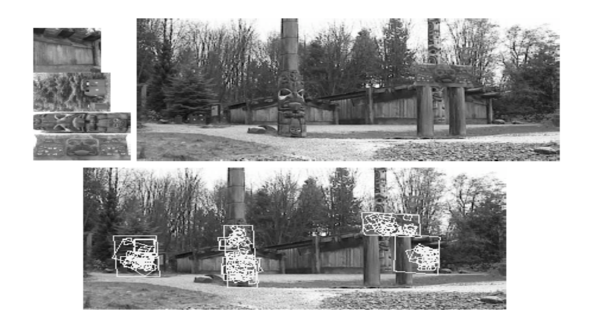
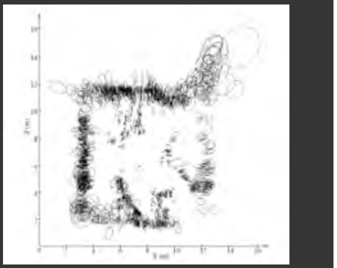


Figure 13: This example shows location recognition within a complex scene. The training images for locations are shown at the upper left and the 640x315 pixel test image taken from a different viewpoint is on the upper right. The recognized regions are shown on the lower image, with keypoints shown as squares and an outer parallelogram showing the boundaries of the training images under the affine transform used for recognition.

Vision-Based Robot Mapping

- FASTSlam innovations
  - Rao-Blackwellized particle filters
- Mapping results for multiple kilometers
- Laser and vision
  - joint issue of IJCV and IJRR prominently vision-based SLAM



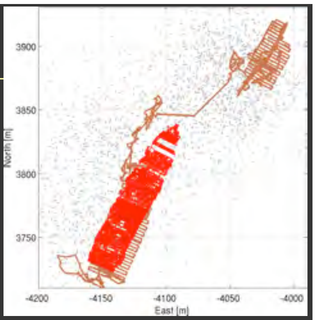
Se, Lowe, Little, 2003

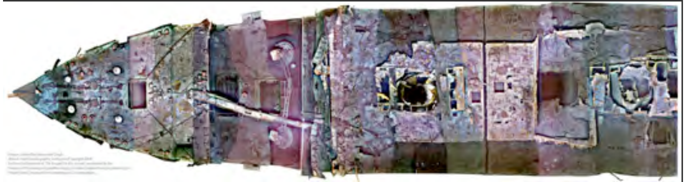
RMS Titanic

Leonard & Eustice

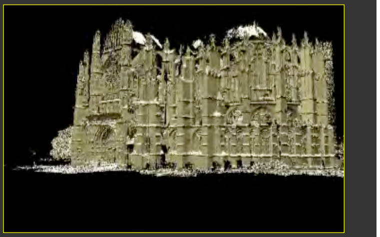
- EKF-based system
- 866 images
- 3494 camera constraints
- Path length 3.1km 2D / 3.4km 3D
- Convex hull > 3100m^2
- 344 min. data / 39 min. ESDF*

*excludes image registration time

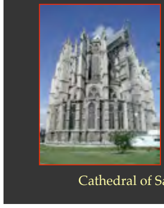


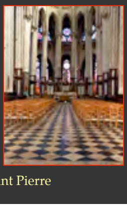


3D Model Building



Reconstruction

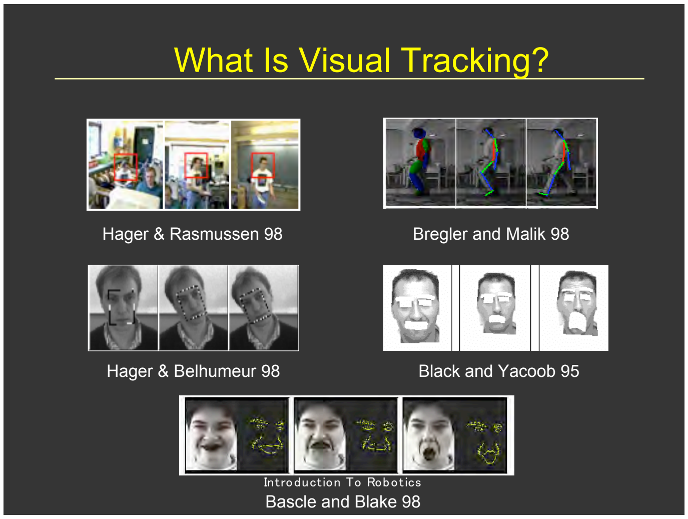




Cathedral of Saint Pierre

(Peter Allen, Columbia University)

VISUAL TRACKING



What Is Visual Tracking?

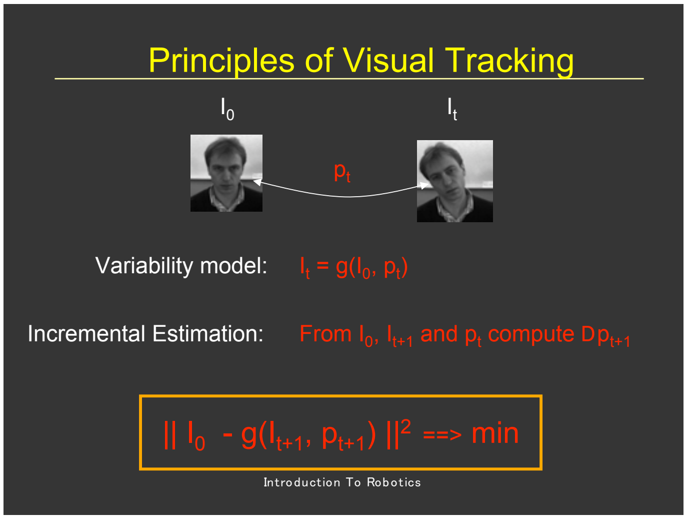
Hager & Rasmussen 98

Bregler and Malik 98

Hager & Belhumeur 98

Black and Yacoob 95

Bascle and Blake 98

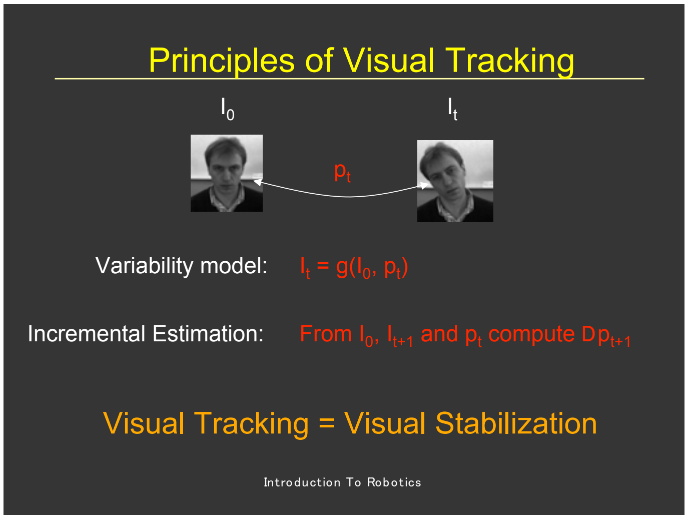


Principles of Visual Tracking

Variability model: $I_t = g(I_0, p_t)$

Incremental Estimation: From I_0, I_{t+1} and p_t compute Dp_{t+1}

$$\boxed{|| I_0 - g(I_{t+1}, p_{t+1}) ||^2 ==> \min}$$

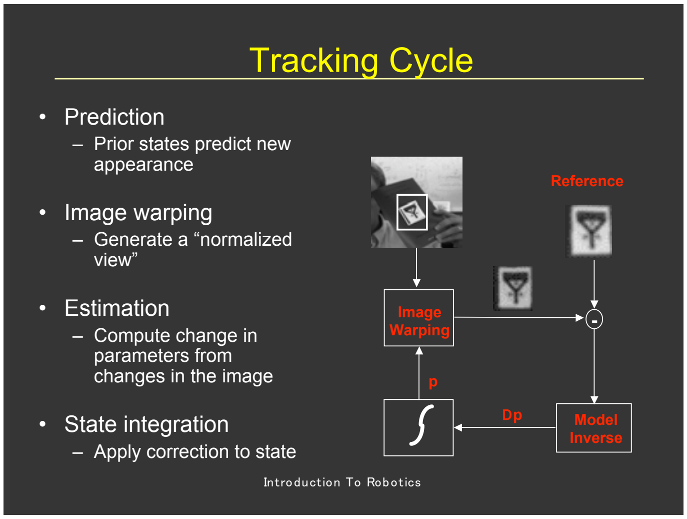


Principles of Visual Tracking

Variability model: $I_t = g(I_0, p_t)$

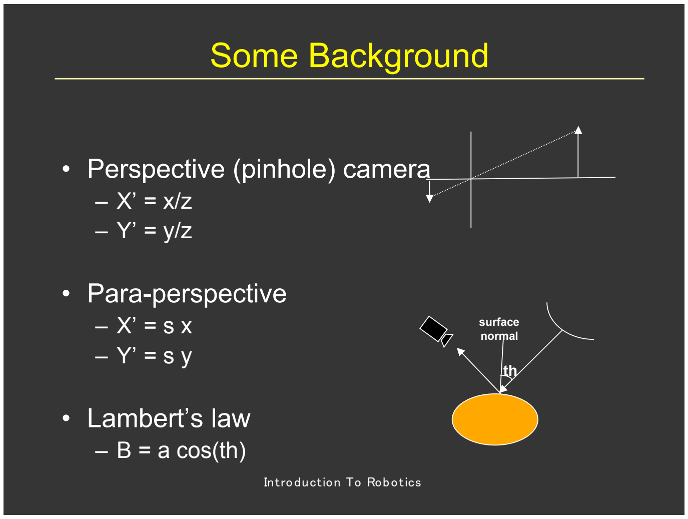
Incremental Estimation: From I_0, I_{t+1} and p_t compute Dp_{t+1}

Visual Tracking = Visual Stabilization



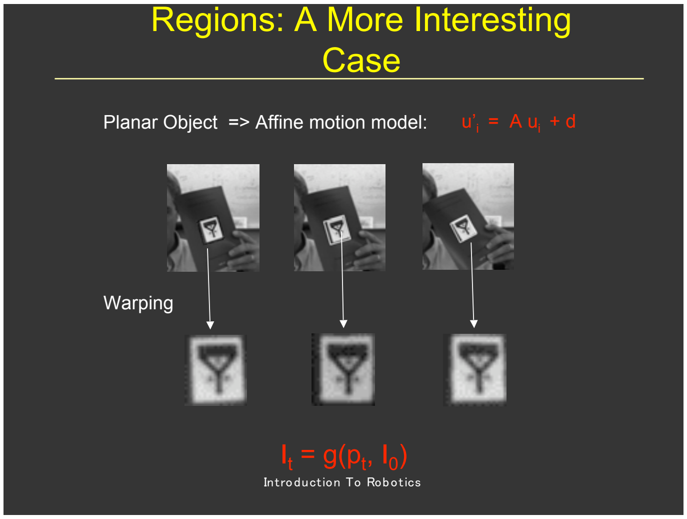
Tracking Cycle

- Prediction
  - Prior states predict new appearance
- Image warping
  - Generate a "normalized view"
- Estimation
  - Compute change in parameters from changes in the image
- State integration
  - Apply correction to state



Some Background

- Perspective (pinhole) camera
  - X' = x/z
  - Y' = y/z
- Para-perspective
  - X' = s x
  - Y' = s y
- Lambert's law
  - B = a cos(th)



Regions: A More Interesting Case

Planar Object => Affine motion model: $u'_i = A u_i + d$

$I_t = g(p_t, I_0)$

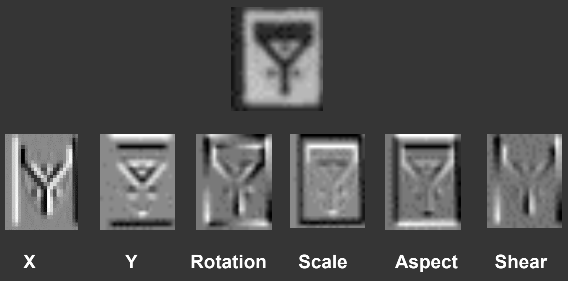
Stabilization Formulation

- Model
  - $I_0 = g(p_t, I_t)$ (image I, variation model g, parameters p)
  - $dI/dt = \mathbf{M}(p_t, I_t)\, dp/dt$ (local linearization $\mathbf{M}$)
- Define an error
  - $e_{t+1} = g(p_t, I_t) - I_0$
- Close the loop
  - $p_{t+1} = p_t - (\mathbf{M}^T \mathbf{M})^{-1} \mathbf{M}^T e_{t+1}$ where $\mathbf{M} = \mathbf{M}(p_t, I_t)$

M is N x m and is time varying!

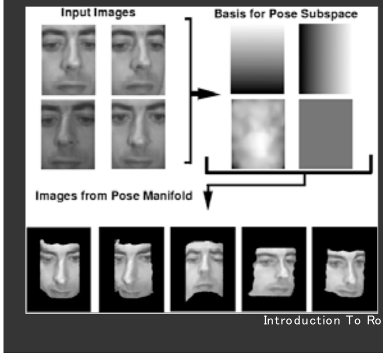
On The Structure of M

Planar Object -> Affine motion model: $u'_i = \mathbf{A}\, u_i + d$



3D Case : Global Geometry

Non-Planar Object: $u_i = \mathbf{A}\, u_i + b\, z_i + d$

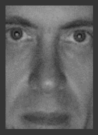


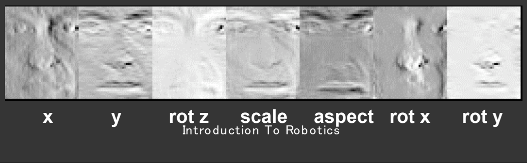
Observations:

- Image coordinates lie in a 4D space
- 3D subspace can be fixed
- Motion in two images give affine structure

3D Case: Local Geometry

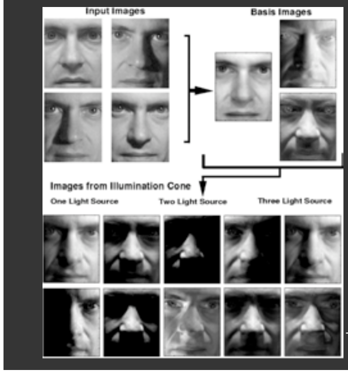
Non-Planar Object: $u_i = \mathbf{A}\, u_i + b\, z_i + d$





3D Case: Illumination Modeling

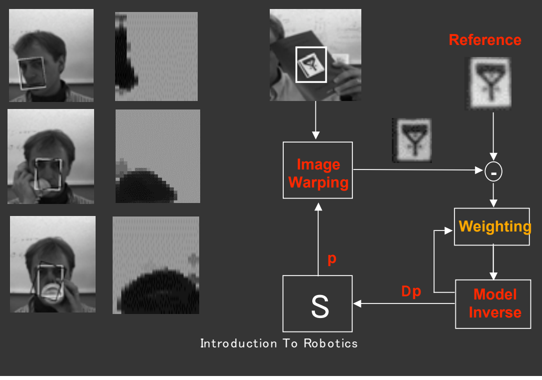
Non-Planar Object: $I_t = \mathbf{B}\, a + I_0$



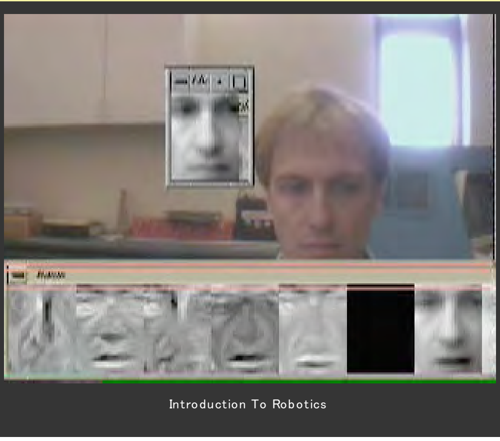
Observations:

- Lambertian object, single source, no cast shadows => 3D image space
- With shadows => a cone
- Empirical evidence suggests 5 to 6 basis images suffices

Handling Occlusion

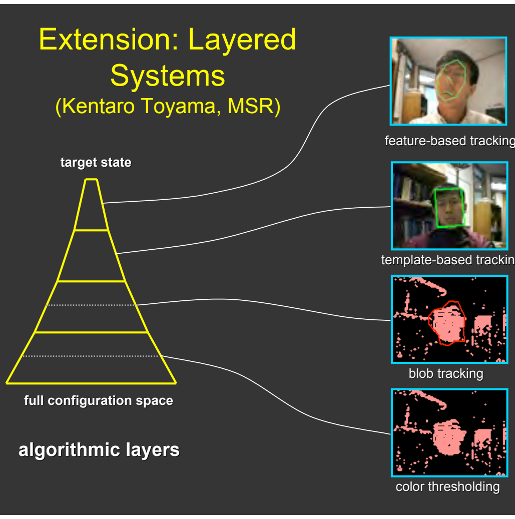


A Complete Implementation



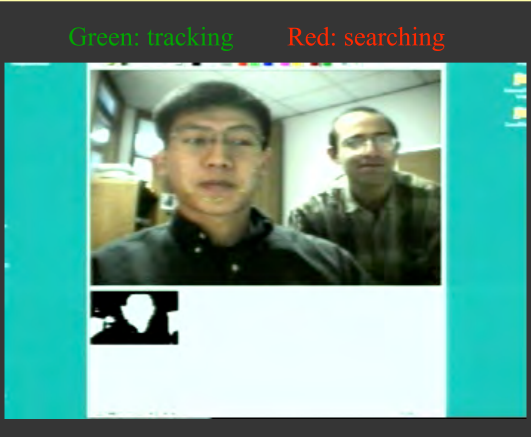
Extension: Layered Systems

(Kentaro Toyama, MSR)

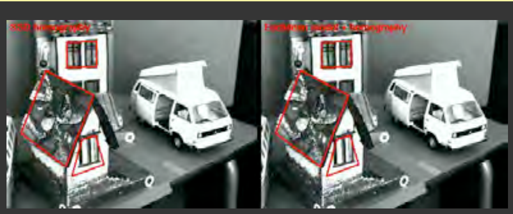


Layered System: Example

Green: tracking Red: searching

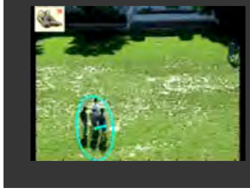


Motion, Tracking, Control



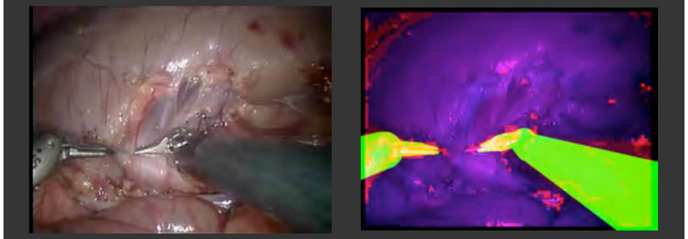
Conventional image-plane SSD 3D SSD

M. Jagersand, U. Alberta



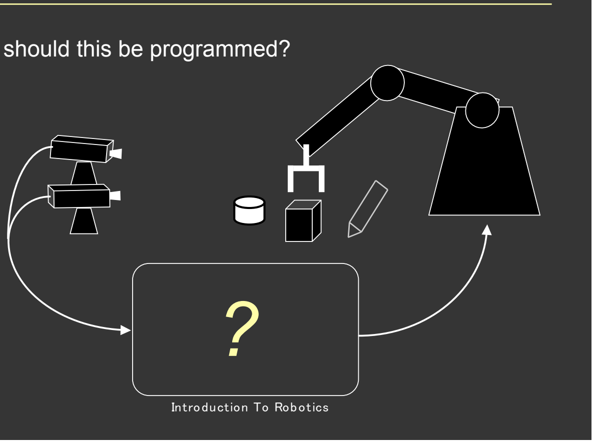
G. Hager, JHU

Adding Kinematics



Vision-Based Control

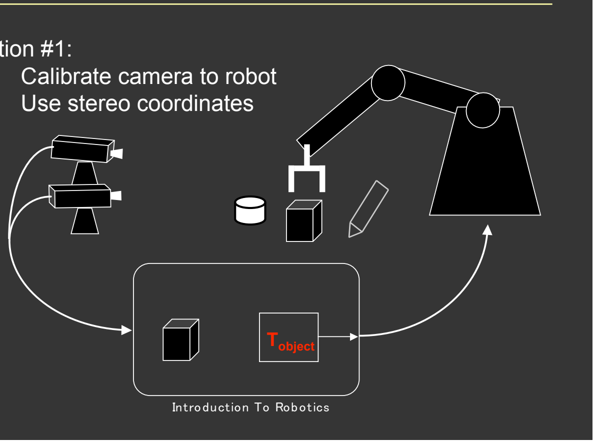
How should this be programmed?



Vision-Based Control

Solution #1:

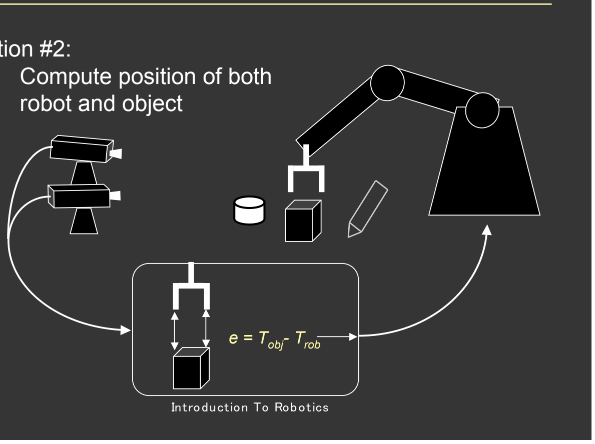
Calibrate camera to robot
Use stereo coordinates



Vision-Based Control

Solution #2:

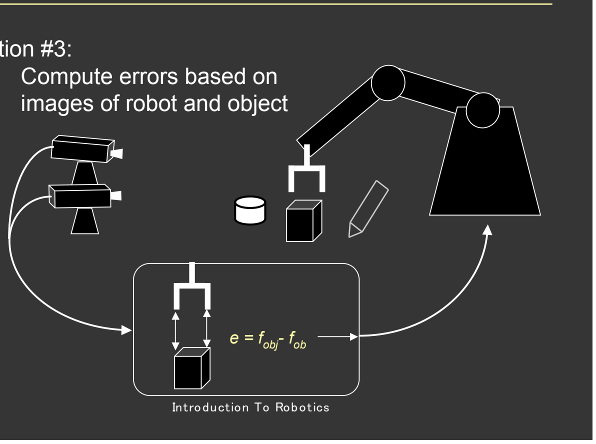
Compute position of both
robot and object



Vision-Based Control

Solution #3:

Compute errors based on
images of robot and object



An Observation

Given:

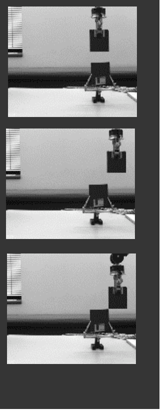
a desired kinematic constraint $T(f_1,f_2) = 0$
an encoding with $e(y_1,y_2) = 0$ iff $T(f_1,f_2) = 0$

Compute:

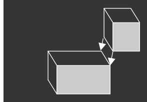
de/dt = J_e dq/dt
dq/dt = - J_e^{-1} e(y_1, y_2)

Result:

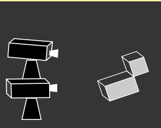
1. If stable, e->0. This implies T->0.
2. Accuracy is *calibration independent*.



More Formally



Task function T
Feature configuration f
Task: T(f) = 0



Set of cameras ***C***
Actual camera **C** є ***C***
Observation **y = C(f)**

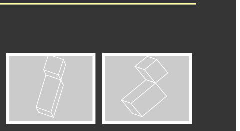


Image encoding **E**
Image features **y**
New task **E(y)** = 0

When can we ensure

$\boxed{T(f) = 0} \longleftrightarrow \boxed{E(y) = 0}$

How can we specify all such tasks?

Example Camera Model Classes

Fix a viewspace **V**

Given C_0 injective on **V**

$\mathbf{C}_{all}$ [C_0] º{ *C* :*C* injective on **V**, Im *C* = Im C_0}

"weakly calibrated injective cameras"

Given projective 2-camera C_0 inj. on **V**

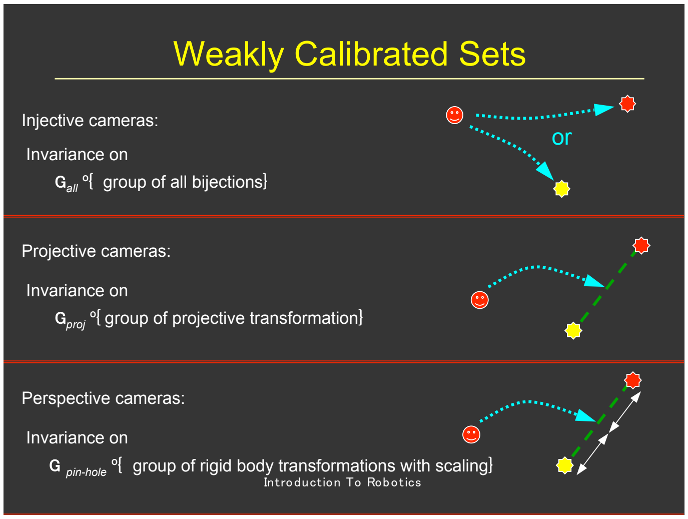
$\mathbf{C}_{proj}$ [C_0] ° **C** $_{all}$[C_0] È { set of all projective 2-camera models}

"weakly calibrated projective cameras"

Given pin-hole 2-camera C_0 inj. on **V**

$\mathbf{C}_{persp}$ [C_0] ° **C** $_{all}$[C_0] È { set of all pin-hole 2-camera models}

"weakly calibrated perspective cameras"



Weakly Calibrated Sets

Injective cameras:

Invariance on

G_{all} °{ group of all bijections}

Projective cameras:

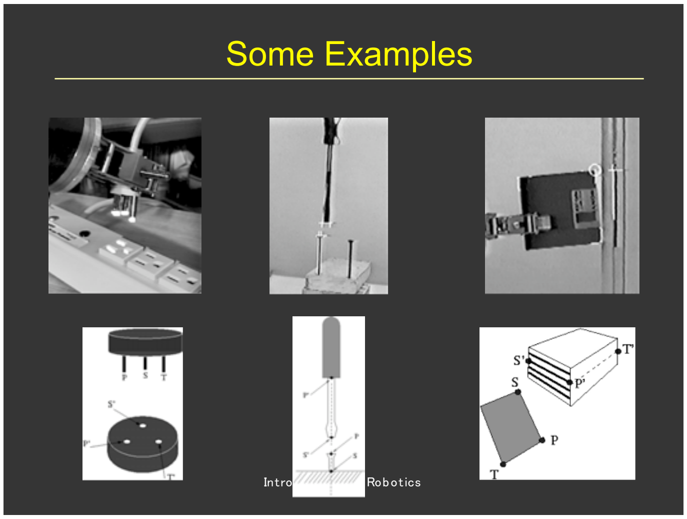
Invariance on

G_{proj} °{ group of projective transformation}

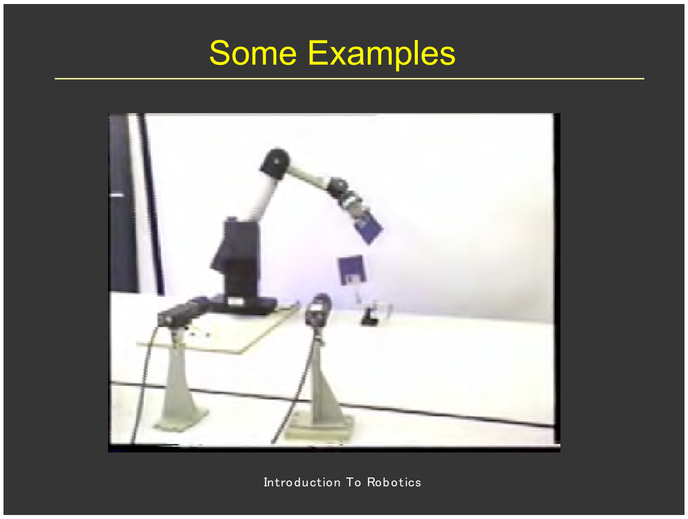
Perspective cameras:

Invariance on

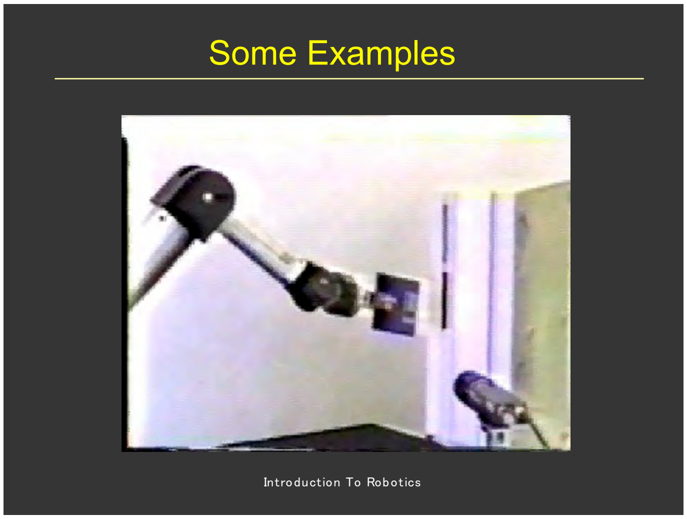
$G_{pin\text{-}hole}$ °{ group of rigid body transformations with scaling}



Some Examples



Some Examples



Some Examples

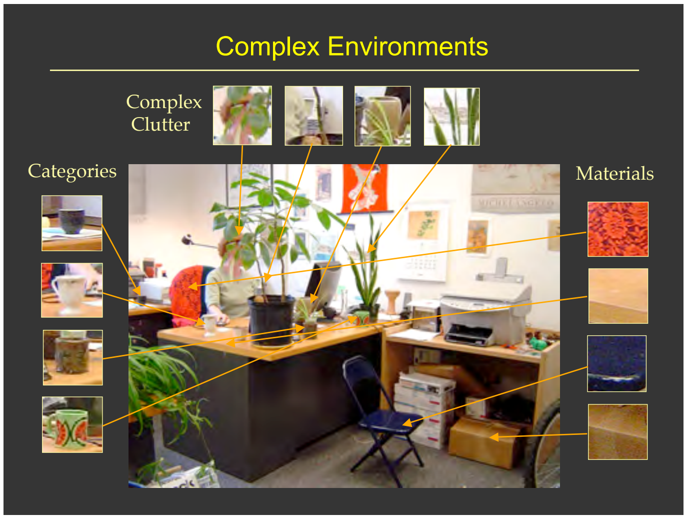
Future Challenges

Complex Geometry

Deformable Objects

Complex Objects

The pieces are starting to appear, why don't we see real systems?



Complex Environments

Complex Clutter

Categories

Materials

Challenge: Highly Dynamic Environments

Recovering Geometry, Egomotion, Individual/Group Trajectories, and Activities

Human Interaction

- Motivators
  - aging population
  - enabling disabled
  - huge market
- Challenges (research)
  - highly integrative
  - unstructured problems
  - adaptivity
- Challenges (market)
  - high initial investment
  - safety/reliability

Generalization and Learning

- Clear value to “data-driven” approaches
- Rapid progress in recent years in
  - dimensional reduction
  - unsupervised modeling
  - supervised methods
- Current methods still do not
  - scale well
  - make use of problem structure
  - cannot be validated

Cross-Cutting Challenges

- Large-scale verification of algorithms
  - data repositories
  - accepted evaluation methodologies
- System integration
  - almost no one has the resources to do it all and do it right
- Facing the real world
  - > 99% reliability
  - manufacturable
  - scalable