Vision-based Controller with various visual features

Takayuki Nishi
FL10_12_01
27th , July, 2010

Introduction

Vision-based controller

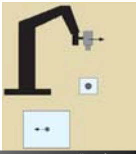
Accomplish the control task using visual information

- Rich information in diversity
- Working under dynamical environments

Rely on techniques from

- Control theory
- Computer vision
- Image processing

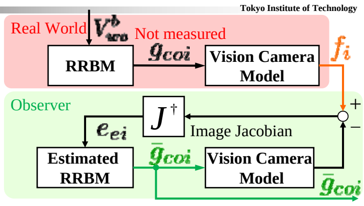
Visual servo control



- Image-based visual servo control (IBVS)
- Position-based visual servo control (PBVS)

Introduction

Visual Motion Observer

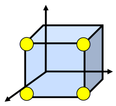


Nonlinear observer to estimate the relative rigid body motion

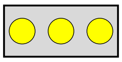
Measured vision data

Target's feature points

Getting more than four three points is required



Feature points



Singular configuration

Objective

Survey about vision-based controller with various features

- Point
- Line
- Luminance

Visual servo control (Classical)

Seth Hutchinson

F. Chaumette and S. Hutchinson, "Visual Servo Control, Part I: Basic Approaches", *IEEE Robotics and Automation Magazine*, Vol. 13, No. 4, Dec., 2006, pp. 82-90.

Controller for constrained feature set

Control objective

Minimize the error

$$e(t) = s(m(t), a) - s^*$$

s : Visual features

s^*: Desired values of s

$m(t)$: Image measurements

a: Parameters that represent potential additional knowledge about the system

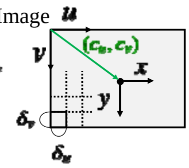


Image coordinates (x, y)

Pixel coordinates (u, v)

Pixel distance δ_u δ_v

Principal point (c_u, c_v)

Image-based visual servo control (IBVS)

s : Set of features

Robust to error in calibration and image noise

Position-based visual servo control (PBVS)

s : Set of 3-D parameters   Required exact pose estimation

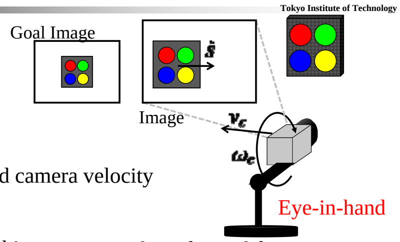
Design of the controller

Camera velocity

$$V_c = (v_c, \omega_c)$$

v_c : Linear velocity

ω_c : Angular velocity



Relation between error and camera velocity

$$\dot{e} = L_e V_c$$

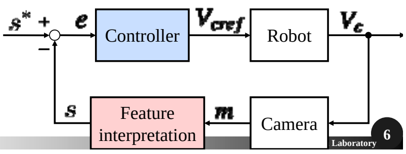
$L_e \in \mathbb{R}^{k \times 6}$: Feature Jacobian k: Number of visual features

Control law $L_e^+ \in \mathbb{R}^{6 \times k}$: Pseudoinverse matrix

$$V_c = -\lambda L_e^+ e$$

The error accomplishes exponential decrease

$$\dot{e} = -\lambda e$$



Derivation of the Jacobian

Image-based visual servo control (IBVS)

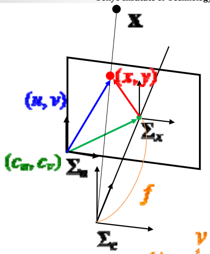
$\mathbf{X} = (X, Y, Z)$: 3-D point in camera frame

$\mathbf{x} = (x, y)$: 2-D point in the image

Perspective projection

$$x = X/Z = \delta_x(u - c_u)/f$$
$$y = Y/Z = \delta_y(v - c_v)/f$$

f : Focal length



Derivative

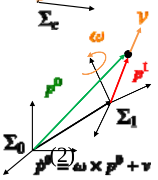
$$\dot{x} = \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z$$
$$\dot{y} = \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{Y} - y\dot{Z})/Z \quad (1)$$

$$\dot{\mathbf{X}} = -v_c - \omega_c \times \mathbf{X}$$

$v_c = [\, v_x \;\; v_y \;\; v_z \,]^T$

$\omega_c = [\, \omega_x \;\; \omega_y \;\; \omega_z \,]^T$

$$\begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases} \quad (2)$$



Example of the Jacobian

$$\dot{x} = \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z$$
$$\dot{y} = \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{Y} - y\dot{Z})/Z \quad (1)$$

$$\begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases} \quad (2)$$

Substitute (2) to (1)

$$\dot{x} = -v_x/Z + xv_z/Z + xy\omega_x - (1 + x^2)\omega_y + y\omega_z$$
$$\dot{y} = -v_y/Z + yv_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \quad (3)$$

From (3), $\dot{\mathbf{x}} = L_e V_c \quad V_c = [\, v_c \;\; \omega_c \,]^T$

$$L_e = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}$$

Z: unknown

Estimation of matrices must be realized in control law

$$V_c = -\lambda \hat{L}_e^+ e$$

SURF (Speeded-Up Robust Features)

SURF (Speeded-Up Robust Features)

- One of feature extraction methods
- Simple and fast computation
- Robust to rotation, scale, and illumination change

Procedure

1. Detection of interest points

   Hessian matrix Find corner of the image

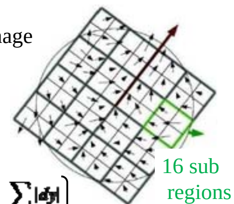
2. Derivate the orientation and features

   Calculate gradient $dx \; dy$

   Approximation with filter

   Feature $16 \times 4 = 64$ dimensions

   Each sub region $\left(\sum dx \;\; \sum |dx| \;\; \sum dy \;\; \sum |dy| \right)$



Visual servo control (Advanced)

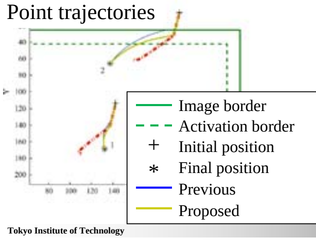

Continuity of Varying-Feature-Set Control Laws

Nicolas Mansard, Anthony Remazeilles, and François Chaumette, Member, IEEE

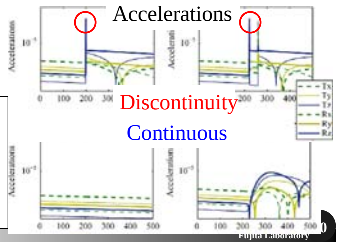
N. Mansard, A. Remazeilles, F. Chaumette, "Continuity of Varying-Feature-Set Control Laws", *IEEE Transactions on Automatic Control*, Vol. 54, No. 11, Nov., 2009, pp. 2493-2505.

F. Chaumette

Point trajectories



Accelerations



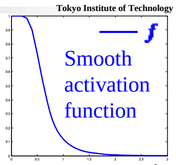
Common idea

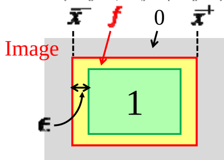
Introduce a buffer area at the activation border

Error function

$$e' = He \qquad H = \mathrm{diag}(h_1, \cdots, h_k)$$

$$h_i = \begin{cases} 1 & \text{if } \; \bar{x}^- + \epsilon \le x \le \bar{x}^+ - \epsilon \\ 0 & \text{if } \; x \ge \bar{x}^+ \text{ or } x \le \bar{x}^- \\ f(x - (\bar{x}^+ - \epsilon)) & \text{if } \; \bar{x}^+ - \epsilon \le x \le \bar{x}^+ \\ f((\bar{x}^- + \epsilon) - x) & \text{if } \; \bar{x}^- \le x \le \bar{x}^- + \epsilon \end{cases}$$





Previous idea $V_c = -\lambda (HL_e)^+ He$

Not continuous : $\mathrm{rank}(HL_e)$ changes

Active feature is less than 3

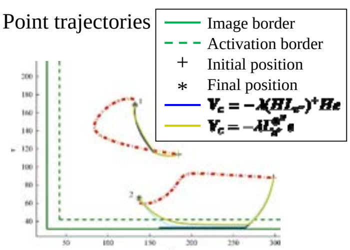
Proposed control law

$V_c = -\lambda L_e^{\oplus H} e \qquad L_e^{\oplus H}$: Continuous inverse of L_e subject to H

- Continuous everywhere
- Act like L_e^+ when $\forall i = 1 \ldots k, h_i \in \{0, 1\}$

Experiment

Point trajectories



Consider both point1 and 2

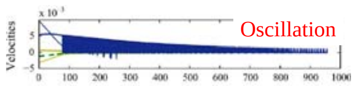
Point2 leave the camera FOV

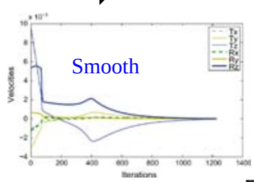
Consider only point1

Point2 enter the camera FOV

Dilemma

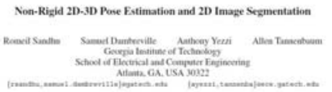
Accelerations of the robot





Previous $V_c = -\lambda (HL_e)^+ He$ Proposed $V_c = -\lambda L_e^{\oplus H} e$

Contour pose estimation


Non-Rigid 2D-3D Pose Estimation and 2D Image Segmentation

Romeil Sandhu Samuel Dambreville Anthony Yezzi Allen Tannenbaum
Georgia Institute of Technology
School of Electrical and Computer Engineering
Atlanta, GA, USA 30322

R. Sandhu, S. Dambreville, A. Yezzi, and A. Tannenbaum. Non-Rigid 2D-3D Pose Estimation and 2D Image Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

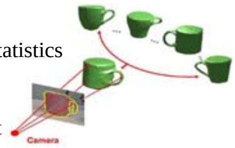
Allen R. Tannenbaum

Pose estimation using contour

Extract the contour curvature using image statistics

- Deal with arbitrary or complex shape

Useful in non-rigid object



- Not need the knowledge of specified 3D model

Pose estimation method

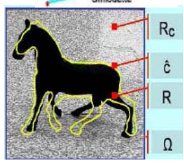
Minimize Energy function

$$E=\int_R r_o(I(\mathbf{x}),\hat{c})d\Omega+\int_{R^c} r_b(I(\mathbf{x}),\hat{c})d\Omega$$

Functions of pose g

$R=\pi(S)$: Region which the surface is projected

R^c : Complementary region $\hat{c}$: Silhouette



$S\subset\mathbb{R}^3$: Surface defining the shape of the object

S_0: Identical reference surface (known)

$S=g(S_0)$ π: Perspective projection $\Omega\in\mathbb{R}^2$: Image domain

r_o r_b : Function measuring the similarity of the image pixels $I(\mathbf{x})$ with a statistical model over the regions R, R^c

(Ex.) $r_o=(I(\mathbf{x})-\mu_o)^2$ $r_b=(I(\mathbf{x})-\mu_b)^2$

μ_o μ_b: Average of image pixels inside(outside) the model object

Experiments

Update procedure Pose parameter set $\xi=[\xi_1,\dots,\xi_6]^T$

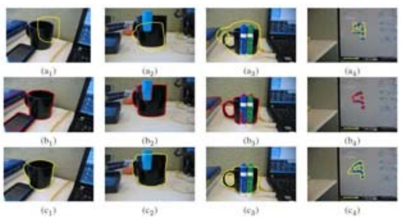
$$\xi(t+\delta t)=\xi(t)-\delta\nabla_\xi E \qquad \frac{\partial E}{\partial \xi_i}=\int_{\hat{c}}(r_o(I(\mathbf{x}))-r_b(I(\mathbf{x})))\left\langle\frac{\partial \hat{c}}{\partial \xi_i},\hat{n}\right\rangle ds$$

Experiments

Initialized contour

Active contour (edge-based)

Image statistics (proposed)



From the experiment

- Robust to occlusion and clutter
- Able to segment different shapes of teacup

Biology pose stabilization


A bio-plausible design for visual attitude stabilization

Andrea Censi, Shuo Han, Sawyer B. Fuller, Richard M. Murray

Andrea Censi, Shuo Han, Sawyer B. Fuller, and Richard M. Murray, "A Bio-Plausible Design for Visual Attitude Stabilization", *IEEE Conference on Decision and Control (CDC), 2009.*

Richard M. Murray

A bio-plausible control

- ➢ Realized on neural substrate
- ➢ Learning
- ➢ Use pixel luminance

Work with uncalibrated camera

Don't use landmark

A new trend in robotics

Control problem

System dynamics

$$\begin{cases}\dot{r}=r(\hat{\omega})\\ I\dot{\omega}=(I\omega)\times\omega+\tau\end{cases}$$ Input torque

$r\in SO(3)$: Attitude

ω : Angular velocity

I : Angular inertia matrix

Visual input (light intensity) at each time

$y(s,t)\triangleq m(r(t)s)$ Unit sphere : $s\in S^2$ map $m: S^2\to\mathbb{R}$

Control problem

Choose the torque τ , such that $y\to g$

Goal image: $g(s)=m(r^*s)$ Goal attitude: $r^*=I$

Proposed controller PD controller

Attitude stabilization (Gradient decent method)

$$J(r) = \frac{1}{2}\|y-g\|_2^2$$

$\omega=\langle g(Sy)\rangle$ stabilize first-order system $\langle f\rangle\triangleq\int_{s\in S^2}f(s)dS$

Estimation of angular velocity

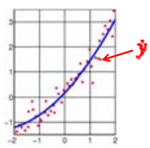
Optic flow equation

$\dot{y}(s,t)=(Sy)^T\omega$ $Sy:=s\times\nabla_s y$ $\dot{s}=\omega\times s$

Least-squares estimation

$\dot{y}=(Sy)^T\omega+\epsilon$ ϵ : Noise

$\hat{\omega}_{LS}\triangleq\left\{\langle(Sy)(Sy)^T\rangle\right\}^{-1}\langle(Sy)\dot{y}\rangle$



Bilinear estimation (less computation)

$\hat{\omega}_{BL}\triangleq c\langle(Sy)\dot{y}\rangle$ $c=(\mathbb{E}_\omega\|\nabla y\|_2^2)^{-1}$: Average of image contrast

Learned Bilinear estimation

$\hat{\omega}_{LBL}\triangleq\langle(\mathbb{E}[M]y)\dot{y}\rangle$ M : Linear operator

Update rule Given training sample $(\omega,\dot{y},y)$

$\dot{M}^k=\alpha(\langle\dot{y}(M^ky)\rangle-\omega^k)\dot{y}y^T$ $\alpha>0$ $k=1,2,3$

M minimizes expected estimation error $\mathbb{E}\{(\langle\dot{y}(M^ky)\rangle-\omega^k)^2\}$

PD controller

PD controller of rigid body dynamics

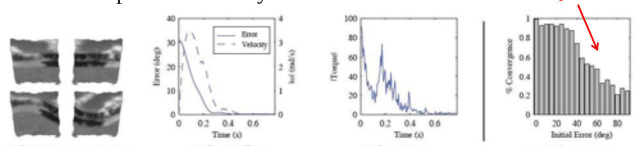
$$\tau = k_p\,(g(\mathbb{E}\{M\}y)) - k_d\,(\dot{g}(\mathbb{E}\{M\}y)) \quad k_p > 0 \;\; k_d > 0$$

Using M in place of S

Simulation (Initial error 30deg)

Visual input : Use fruit fly model

Locally stable



(a) Goal (top) and start (bottom) (b) Rotation Error (c) Control input (d) % of convergence

Conclusion

Vision based controller

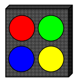
Accomplish the task using visual information

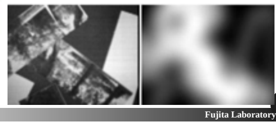
Task

- •Control Robot Motion
- •Pose estimation

Visual information

1. Feature set (point)
2. Contour (line)
3. Luminance



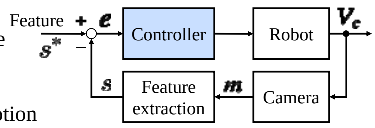
Appendix

Image-based visual servo control (IBVS)

Image-based visual servo control (IBVS)

s : Set of features available in the image data

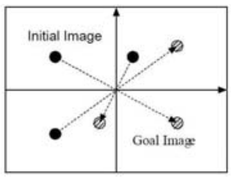
- Robust to error in calibration and image noise
- Local stability
- Unpredictable camera motion



Example of unstable $\hat{L}_e^+ = L_e^+$

Origin symmetry

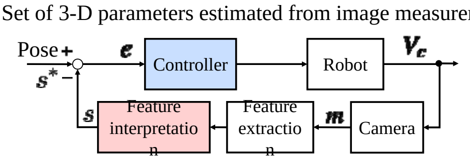
- All of the feature goes to origin
- Camera moves away from the object



Position-based visual servo control (PBVS)

Position-based visual servo control (PBVS)

s : Set of 3-D parameters estimated from image measurements



- Camera moves theoretically
- Required exact pose estimation
- Not consider about image space

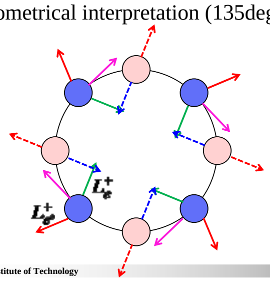
→ Feature goes out of camera image

[Appendix]Approximate the Jacobian

Approximation method

1. $\hat{L}_e^+ = L_e^+$ Calculate L_e
2. $\hat{L}_e^+ = L_{e^*}^+$ L_{e^*}:Value of L_e for the desired position
3. $\hat{L}_e^+ = 1/2(L_e + L_{e^*}^+)$

Geometrical interpretation (135deg rotation)



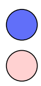


Image motion

- → $\hat{L}_e^+ = L_e^+$
- → $\hat{L}_e^+ = L_{e^*}^+$
- → $\hat{L}_e^+ = 1/2(L_e + L_{e^*}^+)$

Experiment of SURF

Implement SURF

OpenCV (http://opencv.jp/)

- Open source computer vision library for C/C++
  Free download
- Easy to capture the image from USB camera
  Function: cvCreateCameraCapture();
- Easy to use SURF
  Function: cvExtractSURF();

USB camera

Future Works

- Use multiple camera
- How to use CCD camera?

[Appendix2]Example of activation function

Example $V_c = -\lambda (HL_e)^{+} He$

1. Robust Visual Servoing

A. Comport, E. Marchand, F. Chaumette, "Statistically robust 2-D visual servoing", *IEEE Trans. Robot.*, Vol. 22, No. 2, Apr., 2006, pp. 415-420.

$$h_i = \begin{cases} 1 & \text{Full confidence} \\ 0 & \text{The feature is doubtlessly outlier} \end{cases}$$

2. Continuous Visual Servoing Despite Changes of Visibility

N. Garcia-Aracil, E. Malis, R. Aracil-Santonja, and C. Perez-Vidal, "Continuous visual servoing despite the changes of visibility in image features", *IEEE Trans. Robot.*, Vol. 21, No. 6, Apr., 2005, pp. 415-421.

$$H = \sqrt{W} \quad W = \mathrm{diag}(w_1, \cdots, w_k)$$

$$w_i = \begin{cases} 1 & \text{Feature is at the center of the image frame} \\ 0 & \text{Out of the camera field of view (FOV)} \end{cases}$$

[Appendix2]Example of activation function

3. Region reaching control

Bring to a region instead of a point

C. Cheah, D. Wang, and Y. Sun, "Region-reaching control of robots", *IEEE Trans. Robot.*, Vol. 23, No. 6, Dec., 2007, pp. 1260-1264.

The goal region

$$e_i(\mathbf{X}) \le 0 \quad \forall i = 1 \ldots k \qquad \mathbf{X}\text{: Position}$$

Error function

$$e' = H\left[\ 1/2e_1^2 \ \ \ldots \ \ 1/2e_k^2\ \right]^T$$

$$h_i = \begin{cases} 1 & \text{Otherwise} \\ 0 & \text{Corresponding regions have been reached} \end{cases}$$

Control law

$$V_c = -\lambda (HL_e)^T He$$

[Appendix2]Example of activation function

4. Qualitative Servoing

Enlarge the convergence area

A. Remazeilles, N. Mansard, F. Chaumette, "Qualitative visual servoing: application to the visibility constraint", in Proc. *IEEE/RSJ Int. Conf. Robot. Syst. (IROS'06), Beijing, China, on Automatic Control*, Oct., 2006, pp. 4297-4303.

Error function

$$e' = H(e - \bar{e}) \qquad \bar{e}\text{ : Limit of the convergence area}$$

$$h_i = \begin{cases} 1 & \text{System leaves the convergence area} \\ 0 & \text{System enters the convergence area} \end{cases}$$

$$V_c = -\lambda (HL_e)^{+} H(e - \bar{e})$$

[Appendix2]Continuous Inverse

Introduce new operator

Continuous inverse $A^{\ddagger}$ of a matrix A subject to an activation H

$$A \in \mathbb{R}^{k \times n} \quad H = \mathrm{diag}(h_1, \cdots, h_k) \quad h_i \in [0,1]$$

- If $\forall i = 1 \ldots k \ \ h_i \in \{0,1\}$, then
  $$A^{\ddagger H} = (HA)^{+} = (HA)^{+} H$$
- $A^{\ddagger}$ is continuous with regard to H

Implementation of continuous inverse $J^{\oplus H}$

$$J^{\oplus H} = \sum_{\mathcal{P} \in \mathcal{B}(k)} \Big(\prod_{i \in \mathcal{P}} h_i\Big) X_{\mathcal{P}} \qquad X_{\mathcal{P}}\text{: Coupling matrix of } J$$

$$\mathcal{B}(k) = \mathcal{B}(1 \ldots k) = \{\mathcal{P} | \mathcal{P} \subset 1 \ldots k\}$$: All the subsets composed of the k first integer

Proposed control law

$$V_c = -\lambda L_e^{\oplus H} e$$

[Appendix]2 Coupling Matrix

Coupling Matrix

$$\begin{cases} \text{If} & \mathcal{P} = \emptyset \quad X^{\emptyset} = 0_{n \times k} \\ \text{otherwise} & \forall \mathcal{P} \in \mathcal{B}(k)\ X_{\mathcal{P}} = J_{\mathcal{P}}^{+} - \sum_{\mathcal{Q} \subsetneq \mathcal{P}} X_{\mathcal{Q}} \end{cases}$$

$$\mathcal{B}(k) = \mathcal{B}(1 \ldots k) = \{\mathcal{P} | \mathcal{P} \subset 1 \ldots k\}$$

: All the subsets composed of the k first integer

$$J_{\mathcal{P}} = HJ \quad \text{where} \quad h_i = \begin{cases} 1 & i \in \mathcal{P} \\ 0 & \text{otherwise} \end{cases}$$

From the definition

$$J^{\oplus H} = X_{\mathcal{P}} + \sum_{\mathcal{Q} \subsetneq \mathcal{P}} X_{\mathcal{Q}}$$

[Appnendix]Integral Image and filtering

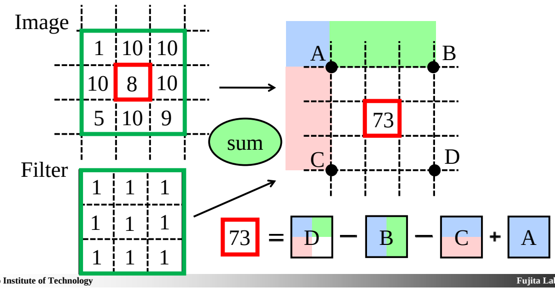
Integral Image $I_\Sigma(\mathbf{x})$

$$I_\Sigma(\mathbf{x}) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j)$$

Pixels : I

Location : $\mathbf{x} = (x,y)^T$

Filtering



[Appendix]Key points detection

Hessian matrix in $\mathbf{x}$ at scale σ

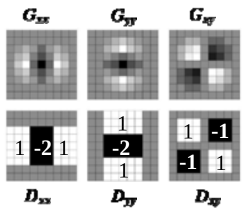
$$\mathcal{H}(\mathbf{x},\sigma) = \begin{bmatrix} L_{xx}(\mathbf{x},\sigma) & L_{xy}(\mathbf{x},\sigma) \\ L_{xy}(\mathbf{x},\sigma) & L_{yy}(\mathbf{x},\sigma) \end{bmatrix}$$

$L_{xx}(\mathbf{x},\sigma)$: Convolution of $\frac{\partial^2}{\partial x^2}G(\sigma)$

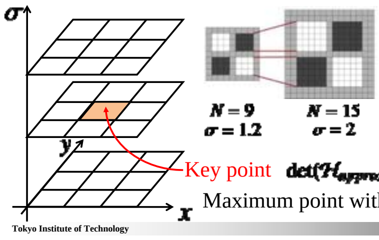
$G(\sigma)$: Gaussian

Approximation with filter

$$\det(\mathcal{H}_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2$$



Apply Hessian detector for various scale

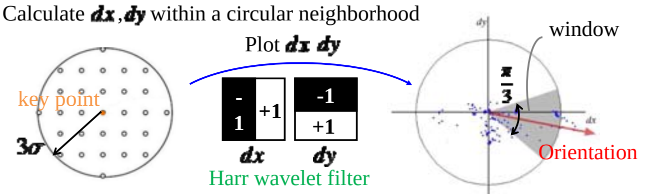


Key point $\det(\mathcal{H}_{approx}) > T$ T : Constant value

Maximum point within 26 neighborhood points

[Appendix]Orientation and Features

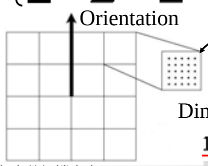
Calculate dx , dy within a circular neighborhood



Features

$$\left(\sum dx \quad \sum|dx| \quad \sum dy \quad \sum|dy|\right)$$

Maximum the sum of the points within the window



16 Sub-region

Sum dx dy $|dx|$ $|dy|$ about 25 points

Dimensions of the feature

$16 \times 4 = 64$

[Appendix3] Calculation

Gradient of the energy function

$$\frac{\partial E}{\partial \xi_i} = \int_\Omega (r_o(I(\mathbf{x})) - r_b(I(\mathbf{x}))\left\langle \frac{\partial \mathbf{e}}{\partial \xi_i}, \hat{\mathbf{n}}\right\rangle d\hat{s}$$

$\mathbf{N} = [\ N_1 \quad N_2 \quad N_3\ ]^T$: Unit normal to S at each point

$$\left\langle \frac{\partial \mathbf{e}}{\partial \xi_i}, \hat{\mathbf{n}}\right\rangle d\hat{s} = \left\langle \frac{\partial \pi(C)}{\partial \xi}, J\frac{\partial \pi(C)}{\partial s}\right\rangle ds$$

$J : \frac{\pi}{2}$ Rotation matrix

$$= \frac{1}{Z^3}\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \begin{bmatrix} 0 & Z & -Y \\ -Z & 0 & X \\ Y & -X & 0 \end{bmatrix}\frac{\partial \mathbf{X}}{\partial s}\right\rangle ds$$

$$= \frac{1}{Z^3}\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \frac{\partial \mathbf{X}}{\partial s}\times \mathbf{X}\right\rangle ds$$

$$= \frac{\|\mathbf{X}\|}{Z^3}\sqrt{\frac{\kappa_X \kappa_t}{K}}\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \mathbf{N}\right\rangle ds$$

$$\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \frac{\partial \mathbf{R}\mathbf{X}_0 + \mathbf{p}}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \mathbf{R}\frac{\partial \mathbf{X}_0}{\partial \xi_i}, \mathbf{R}\mathbf{N}_0\right\rangle = \left\langle \frac{\partial \mathbf{X}_0}{\partial \xi_i}, \mathbf{N}_0\right\rangle$$

K : Gaussian curvature

$$\frac{\partial E}{\partial \xi_i} = \int_\Omega (r_o(I(\mathbf{x})) - r_b(I(\mathbf{x}))\frac{\|\mathbf{X}\|}{Z^3}\sqrt{\frac{\kappa_X \kappa_t}{K}}\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \mathbf{N}\right\rangle ds \qquad \sqrt{\frac{\kappa_X \kappa_t}{K}} \simeq 1$$

κ_X κ_t : Normal curvature in the direction X , t

[Appendix3]Pose estimation method

Translation parameter $\mathbf{T} = [\ p_x \quad p_y \quad p_z\ ]^T = [\ \xi_1 \quad \xi_2 \quad \xi_3\ ]^T$

$$\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \frac{\partial \mathbf{R}\mathbf{X}_0 + \mathbf{P}}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \frac{\partial \mathbf{T}}{\partial \xi_i}, \mathbf{N}\right\rangle = N_i$$

$\mathbf{X} = (X, Y, Z)$: Spatial coordinates related to camera frame

$\mathbf{X}_0$: Points on S_0

Rotation parameter $\mathbf{R} = \exp\left(\begin{bmatrix} \xi_4 \\ \xi_5 \\ \xi_6 \end{bmatrix}^\wedge\right)$

$$\left\langle \frac{\partial \mathbf{X}}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \frac{\partial \mathbf{R}\mathbf{X}_0}{\partial \xi_i}, \mathbf{N}\right\rangle = \left\langle \mathbf{R}\begin{bmatrix} 0 & -\delta_{3,i} & \delta_{2,i} \\ \delta_{3,i} & 0 & -\delta_{1,i} \\ -\delta_{2,i} & \delta_{1,i} & 0 \end{bmatrix}\mathbf{X}_0, \mathbf{N}\right\rangle$$

Update procedure

$$\xi(t + \delta t) = \xi(t) - \delta \nabla_\xi E$$

[Appendix4] System dynamics

System dynamics

$$\begin{cases} \dot{r} = r(\hat{\omega}) \\ \mathbf{I}\dot{\omega} = (\mathbf{I}\omega)\times\omega - \epsilon_\omega\omega + \tau \\ \dot{p} = rv \\ m\dot{v} = mv\times\omega - \epsilon_v v + f \end{cases} \quad (4)$$

Damping term

Body velocity

$$V^b_{ab} = \begin{bmatrix} v^b_{ab} \\ \omega^b_{ab} \end{bmatrix} = \begin{bmatrix} R^T_{ab}\dot{p}_{ab} \\ (R^T_{ab}\dot{R}_{ab})^\vee \end{bmatrix}$$

$$\hat{V}^b_{ab} = g^{-1}_{ab}\dot{g}_{ab}$$

Spatial velocity

$$V^s_{ab} = \begin{bmatrix} v^s_{ab} \\ \omega^s_{ab} \end{bmatrix} = \begin{bmatrix} -\dot{R}_{ab}R^T_{ab}p_{ab} + \dot{p}_{ab} \\ (\dot{R}_{ab}R^T_{ab})^\vee \end{bmatrix}$$

$$\hat{V}^s_{ab} = \dot{g}_{ab}g^{-1}_{ab}$$

$$f = m\ddot{p} = \frac{d}{dt}(mrv^b) = rm\dot{v}^b + \dot{r}mv^b$$

$$\mathbf{I}' = r\mathbf{I}r^T$$

$$\tau = \frac{d}{dt}(\mathbf{I}'\omega^s) = \frac{d}{dt}(r\mathbf{I}r^T\omega^s) = r\mathbf{I}r^T\dot{\omega}^s + \dot{r}\mathbf{I}r^T\omega^s + r\mathbf{I}\dot{r}^T\omega^s = \mathbf{I}'\dot{\omega}^s + \dot{r}r^T\mathbf{I}'\omega^s + \mathbf{I}'r\dot{r}^T\omega^s = \mathbf{I}'\dot{\omega}^s + \omega^s\times\mathbf{I}'\omega^s - \underbrace{\mathbf{I}'\omega^s\times\omega^s}_{0}$$

Body force

$$r^T f = m\dot{v}^b + r^T\dot{r}mv^b$$

$$f^b = m\dot{v}^b + \hat{\omega}_b mv^b = m\dot{v}^b + \omega^b\times mv^b$$

Body torque

$$\mathbf{I}\dot{\omega}^b + \omega^b\times\mathbf{I}\omega^b = \tau^b$$

[Appendix4]Optic flow

Direction $s^i \in \mathbb{S}^2$ $\mathbb{S}^2$:sphere

$$\dot{s} = \mu(s)(I - ss^T)v + \omega \times s$$

$\mu(s)$: inverse of the distance to the object in direction s

$$\nabla_s y \cdot (I - ss^T) = \nabla_s y$$

Chain rule $\dot{y} = \frac{\partial y}{\partial s}\dot{s}$

$$\dot{y}(s,t) = \mu(s)\nabla y(s,t)^T v + (Sy)^T \omega \qquad Sy := s \times \nabla_s y \quad \nabla = \nabla_s$$

[Appendix4]Gradient control law(P controller)

Control the system

$$\begin{cases} \dot{r} = r(\hat{\omega}) \\ \dot{p} = rv \end{cases} \quad (5)$$

Hessian of $J(\varphi, p)$ $\varphi = \log(r)^\vee$

$$C(y,\mu) = \begin{bmatrix} \langle SySy^T \rangle & \langle \mu Sy\nabla y^T \rangle \\ \langle \mu \nabla y Sy^T \rangle & \langle \mu^2 \nabla y \nabla y^T \rangle \end{bmatrix}$$

Cost function

$$J(q) = \frac{1}{2}\int_{s\in\mathbb{S}} (y(s) - g(s))^2 dS = \frac{1}{2}\langle (y-g)^2 \rangle$$

$$\langle f \rangle := \int_{s\in\mathbb{S}^2} f(s) dS$$

Gradient control law

$$\begin{cases} \omega = \langle (Sy)(g-y) \rangle \\ v = \langle \mu(\nabla y)(g-y) \rangle \end{cases}$$

μ : unknown

Locally stabilize the system asymptotically if $C(y,\mu)$ is positive definite

Except : $y(s) = const.$

Approximate gradient (Considered near q_g)

$$\begin{cases} \omega = \langle (Sy)(g-y) \rangle \\ v = \langle \alpha(\nabla y)(g-y) \rangle \end{cases} \quad (6)$$

Locally stabilize the system asymptotically if there exist $\alpha > 0$ such that $\tilde{C}(y,\mu,\alpha)$ is positive definite

$$\tilde{C}(y,\mu,\alpha) = \begin{bmatrix} \langle SySy^T \rangle & \langle (\frac{\alpha}{2} + \frac{\mu}{2}) Sy\nabla y^T \rangle \\ \langle (\frac{\alpha}{2} + \frac{\mu}{2}) \nabla y Sy^T \rangle & \alpha \langle \mu \nabla y \nabla y^T \rangle \end{bmatrix}$$

[Appendix4]PD controller (derivative term)

Control the second order system (4)

PD controller ⟶ Choose τ and f such that stabilize $v, \omega = 0$

Least-square estimates (relation between $\dot{y}$)

$$\hat{\omega}_{LS} = \left(\langle (Sy)(Sy)^T \rangle\right)^{-1} \langle (Sy)\dot{y} \rangle \qquad (v = 0)$$

$$\hat{v}_{LS} = \left(\langle (\mu\nabla y)(\mu\nabla y)^T \rangle\right)^{-1} \langle (\mu\nabla y)\dot{y} \rangle \qquad (\omega = 0)$$

Bilinear form (for simplicity)

$\hat{\omega}_{BL} := \langle (Sy)\dot{y} \rangle$ More bio-plausible

$\hat{v}_{BL} := \langle (\nabla y)\dot{y} \rangle$ Feedback tolerate uncertainties

$\omega^T \hat{\omega}_{BL} \geq 0$ (within 90deg) $v^T \hat{v}_{BL} \geq 0$ ➡ Suffice to estimate

Control law

$$\begin{cases} \tau = -k_d \hat{\omega}_{BL} = -k_d \langle (Sy)\dot{y} \rangle \\ f = -\alpha k_d \hat{v}_{BL} = -\alpha k_d \langle (\nabla y)\dot{y} \rangle \end{cases} \quad (7)$$

$k_d > 0$

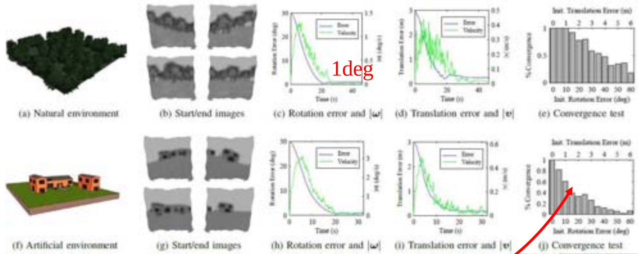
Assume $\tilde{C}(y,\mu,\alpha) > 0$

(7) globally stabilizes ω , v to 0

Simulation

Simulation (Initial error 30deg and 3m)

Visual input : Use fruit fly model



Other facades of the house can be observed

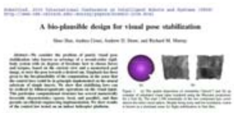
Experiment (another paper)

Helicopter with wireless camera

[Appendix4] Experiment

Experiments Using helicopter with a wireless camera

Shuo Han, Andrea Censi, Andrew D Straw, Richard M Murray, “A bio-plausible design for visual pose stabilization”, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010 (Submitted)



(a) Customized helicopter with wireless cameras

(a) The goal image (before and after processing). (b) Image at time $t \approx 26$ s (before and after processing).

$$J(q) = \frac{1}{2}\int_{s\in\mathbb{S}} (y(s) - g(s))^2 dS = \frac{1}{2}\langle (y-g)^2 \rangle$$

$$\langle f \rangle := \int_{s\in\mathbb{S}^2} f(s) dS$$