A Course on Visual Servo

Koichi Hashimoto
Graduate School of Information Sciences

Menu: Course I

- Basic mathematical tools for control and image processing
- Tools for visual servo: cameras and software
- Image processing basics
- Nonlinear control and robot control
- Basic visual servo

Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- Sampling time issues
- ESM algorithm and visual tracking

Menu: Course I

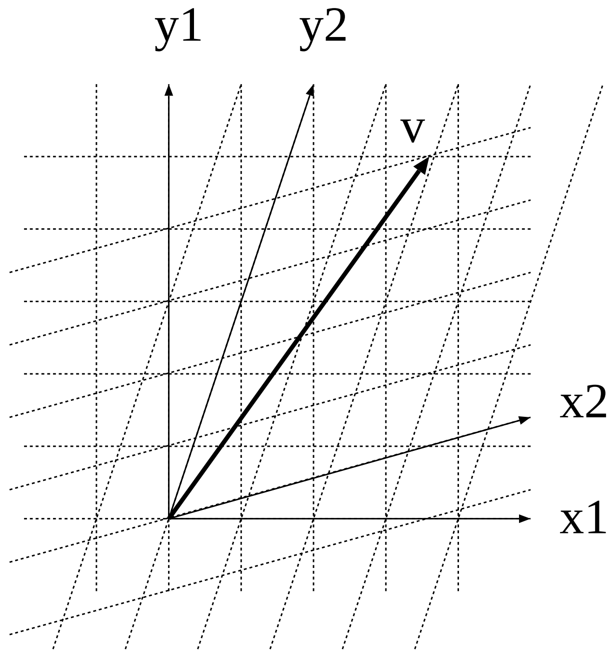
- Basic mathematical tools for control and image processing
- Tools for visual servo: cameras and software
- Image processing basics
- Nonlinear control and robot control
- Basic visual servo

Vector elements

- Vector: direction and length
- Vector length: norm
- Example

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

  – Elements: v_1, v_2?



Coordinate system

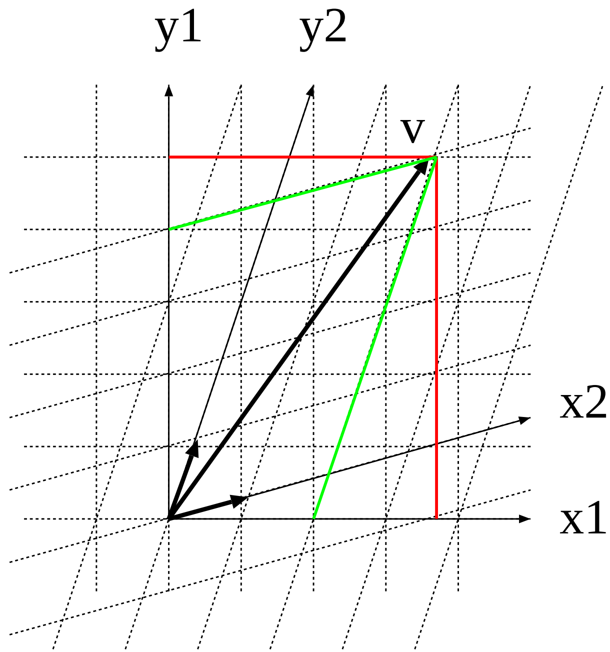
- Vector elements should be associated to a coordinate system
- In $\Sigma_1 = (x_1, y_1)$

$$ {}^{1}\mathbf{v} = \begin{bmatrix} 3.6 \\ 5 \end{bmatrix} $$

- In $\Sigma_2 = (x_2, y_2)$

$$ {}^{2}\mathbf{v} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} $$

- The left upper-script shows the coordinate system in which the elements are expressed



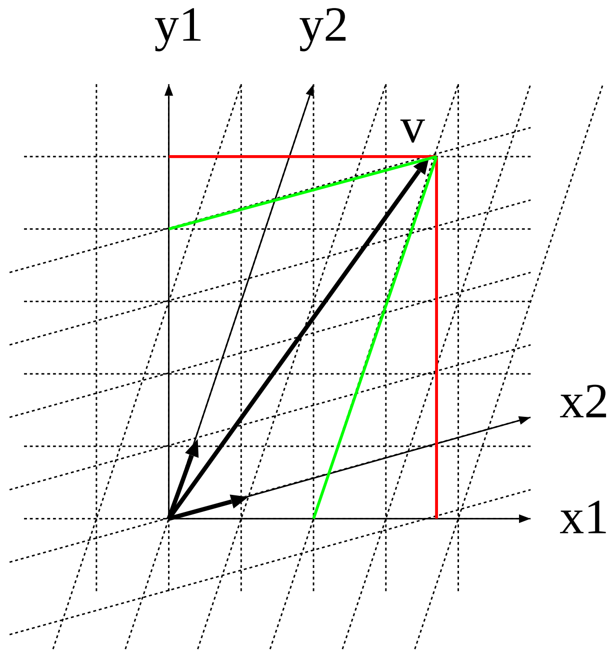
Coordinate transformation I

- Matrix ${}^1\mathbf{R}_2$ is called coordinate transformation matrix.

$$ {}^1\mathbf{v} = {}^1\mathbf{R}_2 {}^2\mathbf{v}, $$

$$ {}^1\mathbf{v} = \begin{bmatrix} 3.6 \\ 5 \end{bmatrix}, \quad {}^2\mathbf{v} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} $$

- How to compute ${}^1\mathbf{R}_2$?



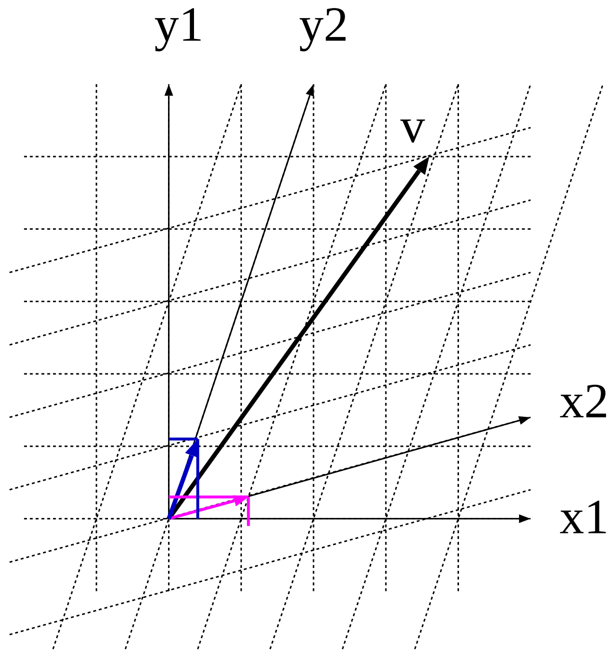
Coordinate transformation II

- Bases of Σ_2: $\mathbf{x}_2, \mathbf{y}_2$
- These vectors have expressions in Σ_1 as follows

$$^1\mathbf{x}_2 = \begin{bmatrix} 1.2 \\ 0.3 \end{bmatrix}, \quad ^1\mathbf{y}_2 = \begin{bmatrix} 0.3 \\ 1.1 \end{bmatrix}$$

- Recall

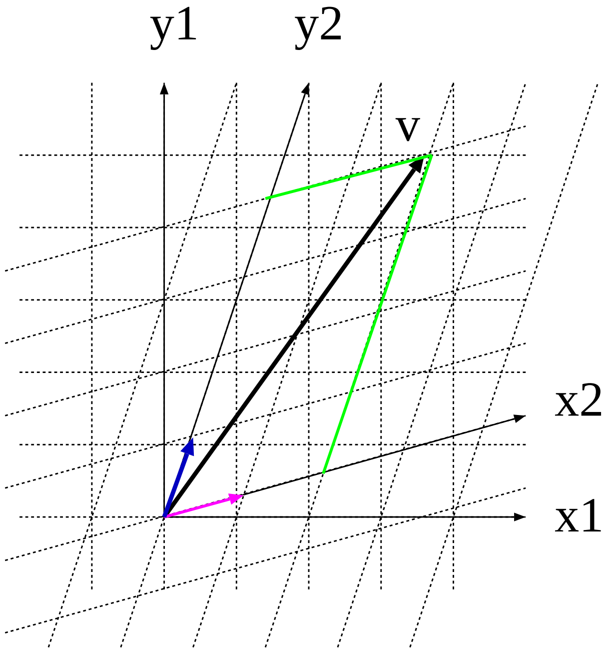
$$^1\mathbf{v} = \begin{bmatrix} 3.6 \\ 5 \end{bmatrix}, \quad ^2\mathbf{v} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$



Coordinate transformation III

- Express $\mathbf{v}$ in Σ_1

$$
\begin{aligned}
{}^1\mathbf{v} &= 2\,{}^1\mathbf{x}_2 + 4\,{}^1\mathbf{y}_2 \\
&= 2\begin{bmatrix} 1.2 \\ 0.3 \end{bmatrix} + 4\begin{bmatrix} 0.3 \\ 1.1 \end{bmatrix} \\
&= \begin{bmatrix} 1.2 & 0.3 \\ 0.3 & 1.1 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \end{bmatrix} \\
\begin{bmatrix} 3.6 \\ 5 \end{bmatrix} &= \begin{bmatrix} 1.2 & 0.3 \\ 0.3 & 1.1 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \end{bmatrix} \\
{}^1\mathbf{v} &= {}^1\mathbf{R}_2\,{}^2\mathbf{v}
\end{aligned}
$$



- Thus we have

$$
{}^1\mathbf{R}_2 = [\,{}^1\mathbf{x}_2 \quad {}^1\mathbf{y}_2\,]
$$

Vector norm, Distance of vectors

- For a vector $\mathbf{v} = [v_1 \ v_2 \ \cdots v_n]^\top$, the norm is given by

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2} = \sqrt{\mathbf{v}^\top \mathbf{v}}$$

- The distance d between two vectors $\mathbf{u}$ and $\mathbf{v}$ is

$$d(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|$$

Image and Kernel

- Consider an $m \times n$ real matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, where $m \geq n$. (Square or Tall matrix)
- Suppose that $\mathbf{M}$ is composed of m column vectors $\mathbf{m}_i$:

$$\mathbf{M} = [\, \mathbf{m}_1 \quad \mathbf{m}_2 \quad \cdots \quad \mathbf{m}_n \,]$$

- When we have

$$\mathbf{y} = \mathbf{M}\mathbf{x}, \quad \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{y} \in \mathbb{R}^m$$

  then the following equation holds

$$\mathbf{y} = x_1\mathbf{m}_1 + x_2\mathbf{m}_2 + \cdots + x_n\mathbf{m}_n$$

  where $\mathbf{x} = [x_1, x_2, \cdots, x_n]^\top$.

Image and Kernel II

$$\mathbf{y} = x_1\mathbf{m}_1 + x_2\mathbf{m}_2 + \cdots + x_n\mathbf{m}_n$$

- $\mathbf{y}$ is a linear combination of $\mathbf{m}_i$.
- By changing $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y}$ moves in a space spanned by $\mathbf{m}_i$.
- This space is called the **image** of $\mathbf{M}$ and denoted by $\mathrm{Im}\mathbf{M}$.

$$\mathrm{Im}\mathbf{M} = \{\mathbf{y} : \mathbf{y} = \mathbf{M}\mathbf{x} \quad \forall\mathbf{x} \in \mathbb{R}^n\}$$

- Suppose that we have $\mathbf{y} = \mathbf{0}$ for some $\mathbf{x} = \mathbf{x}_i$.
- The space spanned by these vectors are called **kernel** of $\mathbf{M}$ and denoted by $\mathrm{Ker}\mathbf{M}$.

$$\mathrm{Ker}\mathbf{M} = \{\mathbf{x} : \mathbf{M}\mathbf{x} = \mathbf{0}\} = \{\mathbf{x} : \mathbf{m}_i^\top\mathbf{x} = 0, \quad i = 1, \ldots, n\}$$

Rank of a matrix (full rank)

- Suppose that all column vectors are linear independent, i.e., suppose that if

$$a_1\mathbf{m}_1 + a_2\mathbf{m}_2 + \cdots + a_n\mathbf{m}_n = \mathbf{0}$$

then we have only one solution

$$a_1 = a_2 = \cdots = a_n = 0.$$

In this case the rank of $\mathbf{M}$ is n and the matrix is called **full rank**.

Rank of a matrix

- If the matrix is not full rank then select $n-1$ vectors from $\mathbf{m}_i$ and check whether they are linear independent or not.
- If the maximum number of linear independent vectors is r then the **rank** of **M** is r and written as

$$\text{rank}\mathbf{M} = r$$

Matrix inverse

- If $\mathbf{M}$ is square and full rank then it has its inverse $\mathbf{M}^{-1}$.

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$$

- If $\mathbf{M}$ is not full rank then it does not have its inverse.

Example

- Consider a set of linear equations

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3 \end{aligned}$$

- Suppose that a_{ij}, b_i $(i, j = 1, 2, 3)$ are known and we want to find $x_i (i = 1, 2, 3)$ to satisfy these equations.

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

- If $\mathbf{A}$ is full rank then the solution is

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Example

- If $\mathbf{A}$ is not full rank, i.e., if

$$\mathbf{a}_1 = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \end{bmatrix}, \; \mathbf{a}_2 = \begin{bmatrix} a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}, \; \mathbf{a}_3 = \begin{bmatrix} a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

  are linear dependent, then how to find the best solution?

- Suppose that $\mathbf{a}_3 = \mathbf{a}_1 + \mathbf{a}_2$. Then the original equation can be re-written as follows:

$$\begin{aligned} \mathbf{b} &= x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + x_3\mathbf{a}_3 = (x_1 + x_3)\mathbf{a}_1 + (x_2 + x_3)\mathbf{a}_2 \\ &= [\mathbf{a}_1 \quad \mathbf{a}_2] \begin{bmatrix} x_1 + x_3 \\ x_2 + x_3 \end{bmatrix} \\ \mathbf{b} &= \bar{\mathbf{A}}\bar{\mathbf{x}} \qquad \text{(3 equations but 2 unknowns)} \end{aligned}$$

Example

- New system

$$\mathbf{b} = \bar{\mathbf{A}}\bar{\mathbf{x}}$$

- The least square solution of this equation is defined by $\bar{\mathbf{x}}$ that minimizes

$$J = \|\bar{\mathbf{A}}\bar{\mathbf{x}} - \mathbf{b}\| = (\bar{\mathbf{A}}\bar{\mathbf{x}} - \mathbf{b})^\top(\bar{\mathbf{A}}\bar{\mathbf{x}} - \mathbf{b})$$

- It must satisfy

$$\frac{\partial J}{\partial \bar{\mathbf{x}}} = 2\bar{\mathbf{x}}^\top\bar{\mathbf{A}}^\top\bar{\mathbf{A}} - 2\mathbf{b}^\top\bar{\mathbf{A}} = \mathbf{0}$$

i.e.,

$$\bar{\mathbf{A}}^\top\bar{\mathbf{A}}\bar{\mathbf{x}} - \bar{\mathbf{A}}^\top\mathbf{b} = \mathbf{0}$$

and we have

$$\bar{\mathbf{x}} = (\bar{\mathbf{A}}^\top\bar{\mathbf{A}})^{-1}\bar{\mathbf{A}}^\top\mathbf{b}$$

Example

- How to obtain $\mathbf{x}$ from $\bar{\mathbf{x}}$?
- Minimum norm condition ($\min s$):

$$s = x_1^2 + x_2^2 + x_3^2 = (\bar{x}_1 - x_3)^2 + (\bar{x}_2 - x_3)^2 + x_3^2$$

i.e.,

$$\frac{ds}{dx_3} = -2\bar{x}_1 - 2\bar{x}_2 + 6x_3 = 0, \quad x_3 = \frac{1}{3}(\bar{x}_1 + \bar{x}_2)$$

and we have

$$x_1 = \frac{2}{3}\bar{x}_1 - \frac{1}{3}\bar{x}_2,\ x_2 = \frac{2}{3}\bar{x}_2 - \frac{1}{3}\bar{x}_1,\ x_3 = \frac{1}{3}\bar{x}_1 + \frac{1}{3}\bar{x}_2$$

- This solution minimizes

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\| \quad \text{as well as} \quad \|\mathbf{x}\|$$

Generalized Inverse

- Linear equation

$$\mathbf{M}\mathbf{x} = \mathbf{y}$$

- If $\mathbf{M}$ is tall and full rank, $(\mathbf{M}^\top\mathbf{M})$ becomes square and full rank and there exists $(\mathbf{M}^\top\mathbf{M})^{-1}$.
- The generalized inverse

$$\mathbf{M}^\dagger = (\mathbf{M}^\top\mathbf{M})^{-1}\mathbf{M}^\top$$

satisfies a solution that minimizes

$$\mathbf{x} = \operatorname*{argmin}_{\mathbf{x}} \|\mathbf{M}\mathbf{x} - \mathbf{y}\| = \mathbf{M}^\dagger\mathbf{y}$$

- The matrix also satisfies

$$\mathbf{M}\mathbf{M}^\dagger\mathbf{M} = \mathbf{M}, \quad \mathbf{M}^\dagger\mathbf{M}\mathbf{M}^\dagger = \mathbf{M}^\dagger$$

Singular Value Decomposition I

- Very important and useful decomposition!
- Given a $m \times n$ $(m \geq n)$ real matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$.
- Suppose that the rank of $\mathbf{M}$ is r $(< n)$.
- SVD

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

where

$$\begin{aligned}
\mathbf{U} &= [\,\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_m\,] \in \mathbb{R}^{m \times m} \\
\mathbf{\Sigma} &= \begin{bmatrix} \sigma_1 & & & & & \\ & \sigma_2 & & & & \\ & & \ddots & & & \\ & & & \sigma_r & & \\ & & & & \ddots & \\ & & & & & \sigma_n \end{bmatrix} \in \mathbb{R}^{m \times n} \\
\mathbf{V}^\top &= [\,\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_n\,]^\top \in \mathbb{R}^{n \times n}
\end{aligned}$$

Singular Value Decomposition II

- SVD

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

where

$$\begin{aligned} \mathbf{U}^\top\mathbf{U} &= \mathbf{U}\mathbf{U}^\top = \mathbf{I}, \quad \mathbf{V}^\top\mathbf{V} = \mathbf{V}\mathbf{V}^\top = \mathbf{I}, \\ \sigma_1 &\geq \sigma_2 \geq \cdots \geq \sigma_r > 0, \quad \sigma_{r+1} = \cdots = \sigma_n = 0 \end{aligned}$$

- Singular value: σ_i $(i = 1, \ldots, n)$

Property of SVD I

- $\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$

$$\mathbf{V}^\top \mathbf{v}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow i, \quad \boldsymbol{\Sigma} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \sigma_i \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \mathbf{U} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \sigma_i \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \sigma_i \mathbf{u}_i$$

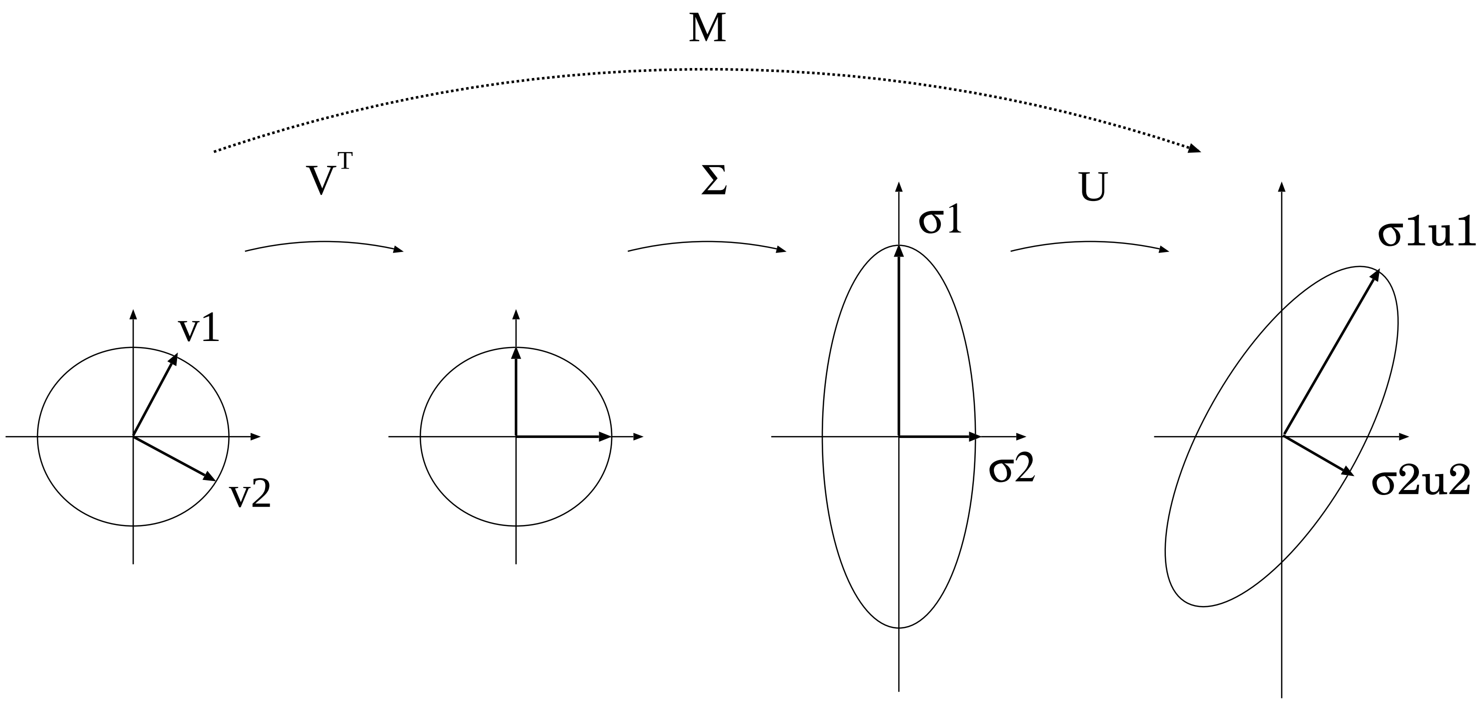
- Thus

$$\mathbf{M}\mathbf{v}_i = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top \mathbf{v}_i = \sigma_i \mathbf{u}_i$$

- Matrix $\mathbf{M}$ rotates the unit vector $\mathbf{v}_i$ by $\mathbf{V}^\top$, and magnify σ_i and rotate by $\mathbf{U}$; and we obtain $\sigma_i \mathbf{u}_i$.

Property of SVD II

- Since $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$,
- σ_1 and $\mathbf{v}_1$ are called maximum singular value and maximum singular vector, respectively.
- The ratio σ_1/σ_n is called condition number and plays an important roll in numerical calculation.



Property of SVD III

- Generalized inverse of non-full rank matrix

$$\mathbf{M}^{\dagger} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{\top}$$

where

$$\mathbf{\Sigma}^{-1} = \begin{bmatrix} \sigma_1^{-1} & & & & & & \\ & \sigma_2^{-1} & & & & & \\ & & \ddots & & & & \\ & & & \sigma_r^{-1} & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix}$$

Example of SVD I

- **Problem:** Given $\mathbf{M} \in \mathbb{R}^{m \times n}$ $(m > n)$, find $\mathbf{x} \in \mathbb{R}^n$ that satisfies

$$\mathbf{M}\mathbf{x} = \mathbf{0} \quad \text{and} \quad \|\mathbf{x}\| = 1$$

- When rank of $\mathbf{M}$ is r $(< n)$, dimension of Ker$\mathbf{M}$ is $n - r$.
- The solution should be in Ker$\mathbf{M}$.

$$\forall \mathbf{x} \in \text{Ker}\mathbf{M}, \quad \|\mathbf{x}\| = 1$$

Example of SVD I

- Singular value decomposition of $\mathbf{M}$:

$$\mathbf{M}\mathbf{v}_i = \sigma_i \mathbf{u}_i, \quad \|\mathbf{v}_i\| = 1$$

  and $\sigma_i = 0$ $(i = r+1, \ldots, n)$.
- Thus $\mathbf{v}_i$ $(i = r+1, \ldots, n)$ are the vectors that span Ker$\mathbf{M}$.
- **Solution:**

$$\mathbf{x} = \sum_{i=r+1}^{n} \alpha_i \mathbf{v}_i \quad \text{where} \quad \sum_{i=r+1}^{n} |\alpha_i| = 1$$

Example of SVD II

- **Problem:** Let $\mathbf{e} = \mathrm{M}\mathbf{x}$ and find the solution that satisfy

$$\|\mathbf{e}\| = \|\mathrm{M}\mathbf{x}\| \to \text{min} \quad \text{and} \quad \|\mathbf{x}\| = 1$$

- SVD of M and find $\mathbf{v}_i, (i = 1, \ldots, n)$. Then $\mathbf{x}$ should be expressed by

$$\mathbf{x} = x_1\mathbf{v}_1 + \cdots + x_n\mathbf{v}_n, \quad x_1^2 + \cdots + x_n^2 = 1$$

- Then we have

$$\mathbf{e} = \mathrm{M}\mathbf{x} = \sigma_1 x_1 \mathbf{u}_1 + \cdots + \sigma_n x_n \mathbf{u}_n$$

Example of SVD II

- Since $\mathbf{u}_i$ $(i = 1, \ldots, n)$ are orthonormal vectors, the norm of error vector is evaluated by

$$\|e\| = (\sigma_1 x_1)^2 + \cdots + (\sigma_n x_n)^2, \quad \sigma_1 \geq \cdots \sigma_n > 0$$

- To minimize the norm $\|\mathbf{e}\|$, it should be

$$x_1 = \cdots = x_{n-1} = 0, x_n = 1$$

- **Solution:** Thus we finally obtain

$$\mathbf{x} = \mathbf{v}_n$$

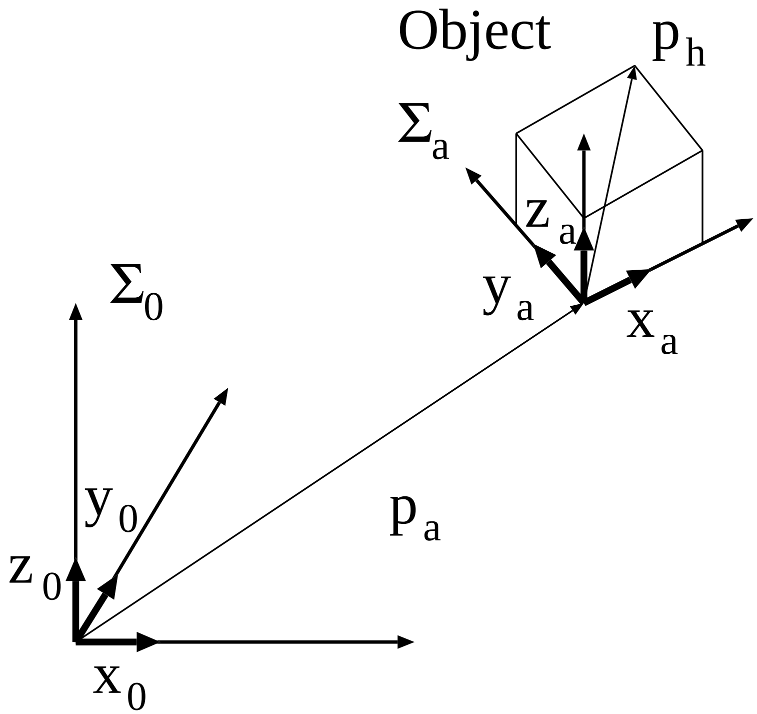
- **(Important!)** The solution of

$$\|\mathbf{e}\| = \|\mathbf{M}\mathbf{x}\| \to \text{min}, \quad \|\mathbf{x}\| = 1$$

is $\mathbf{x} = \mathbf{v}_n$.

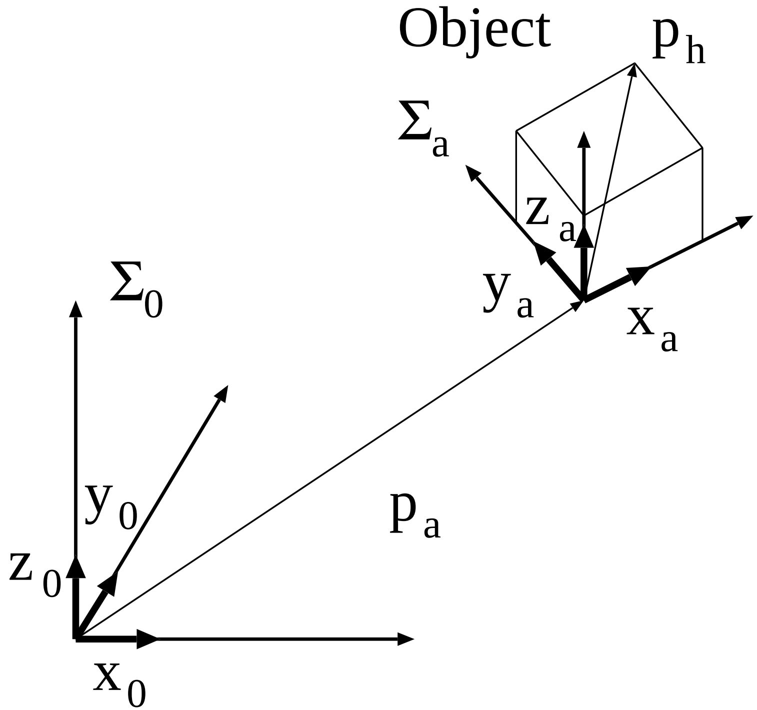
Coordinate system I

- Base coordinate system: Σ_0
- Object coordinate system: Σ_a
- Position of the object: $\mathbf{p}_a$
- Orientation of the object: $\mathbf{R} = [\,\mathbf{x}_a \quad \mathbf{y}_a \quad \mathbf{z}_a\,]$



Coordinate system II

- A point on the object is given by a constant vector $\mathbf{p}_h$



Coordinate system II

- Point vector in Σ_a: $\mathbf{p}_h = [x_h, y_h, z_h]^\top$
- In Σ_a system:

$$\mathbf{p}_h = \begin{bmatrix} x_h \\ y_h \\ z_h \end{bmatrix} = x_h \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + y_h \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + z_h \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

- In Σ_0 system:

$$\begin{aligned} {}^0\mathbf{p}_h &= x_h \mathbf{x}_a + y_h \mathbf{y}_a + z_h \mathbf{z}_a \\ &= [\mathbf{x}_a \quad \mathbf{y}_a \quad \mathbf{z}_a] \begin{bmatrix} x_h \\ y_h \\ z_h \end{bmatrix} \\ &= \mathbf{R}\mathbf{p}_h = {}^0\mathbf{R}_a {}^a\mathbf{p}_h \end{aligned}$$

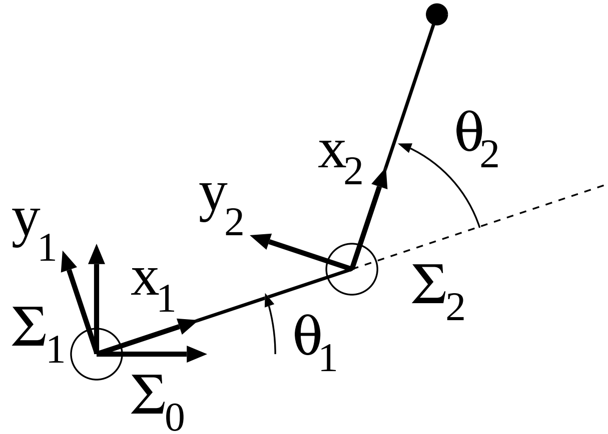
2 Link robot I

- In Σ_1, tip of link 1

$$^{1}\mathbf{p}_2 = \begin{bmatrix} l_1 \\ 0 \\ 0 \end{bmatrix}$$

- In Σ_0, tip of link 1

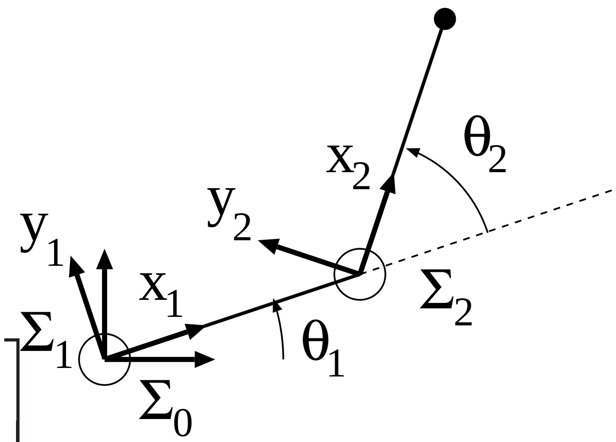
$$^{0}\mathbf{p}_2 = {}^{0}\mathbf{R}_1(\theta_1) \begin{bmatrix} l_1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} l_1 \cos\theta_1 \\ l_1 \sin\theta_1 \\ 0 \end{bmatrix}$$



2 Link robot II

- In Σ_2, tip of link 2

$$^2\mathbf{p}_e = \begin{bmatrix} l_2 \\ 0 \\ 0 \end{bmatrix}$$



- In Σ_1, tip of link 2

$$^1\mathbf{p}_e = {}^1\mathbf{R}_2{}^2\mathbf{p}_e + {}^1\mathbf{p}_2 = \begin{bmatrix} l_1 + l_2\cos\theta_2 \\ l_2\sin\theta_2 \\ 0 \end{bmatrix}$$

2 Link robot II

- In Σ_0, tip of link 2

$$\begin{aligned}
{}^0\mathbf{p}_e &= {}^0\mathbf{R}_1 {}^1\mathbf{p}_e = {}^0\mathbf{R}_1 \begin{bmatrix} l_1 + l_2 \cos\theta_2 \\ l_2 \sin\theta_2 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \\ l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \\ 0 \end{bmatrix}
\end{aligned}$$

- Orientation of link 2

$$ {}^0\mathbf{R}_2 = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) & 0 \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

Homogeneous transformation

- Suppose that $[x, y, z]^\top$ and $[x, y, z, 1]^\top$ are identical.
- Also $[x, y, z, w]^\top$ and $[x/w, y/w, z/w, 1]^\top$ are identical.

- 2 link robot example:

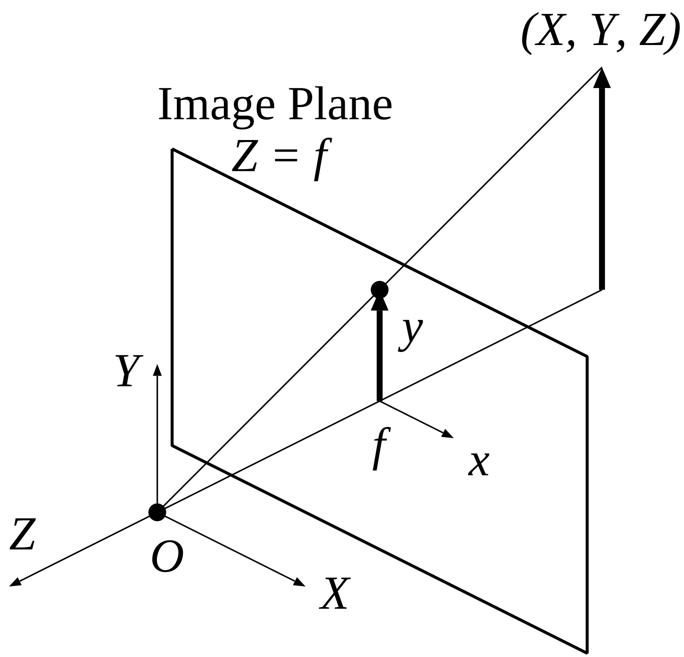
$$\begin{bmatrix} {}^1\mathbf{p}_e \\ 1 \end{bmatrix} = \begin{bmatrix} {}^1\mathbf{R}_2 & {}^1\mathbf{p}_e \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^2\mathbf{p}_e \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^0\mathbf{p}_e(t) \\ 1 \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{R}_1(t) & {}^0\mathbf{p}_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1\mathbf{R}_2(t) & {}^1\mathbf{p}_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^2\mathbf{p}_e \\ 1 \end{bmatrix}$$

Example

- Point at $[X \ \ Y \ \ Z]^\top$ in camera coordinate system is projected to

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



Example

- In homogeneous transformation

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Actually we have

$$sx = fX, \quad sy = fY, \quad s = Z$$

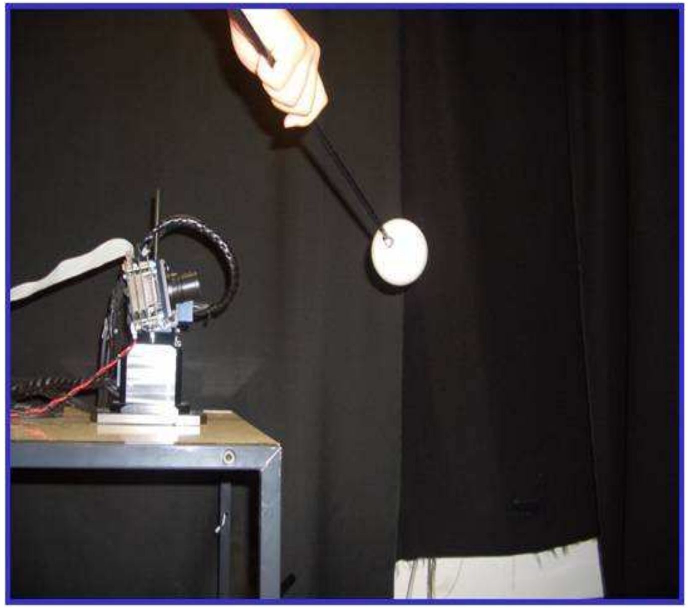
and

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

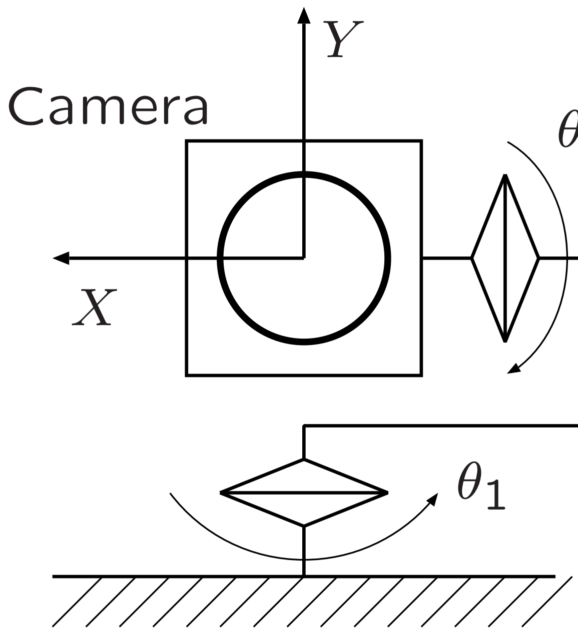
Menu: Course I

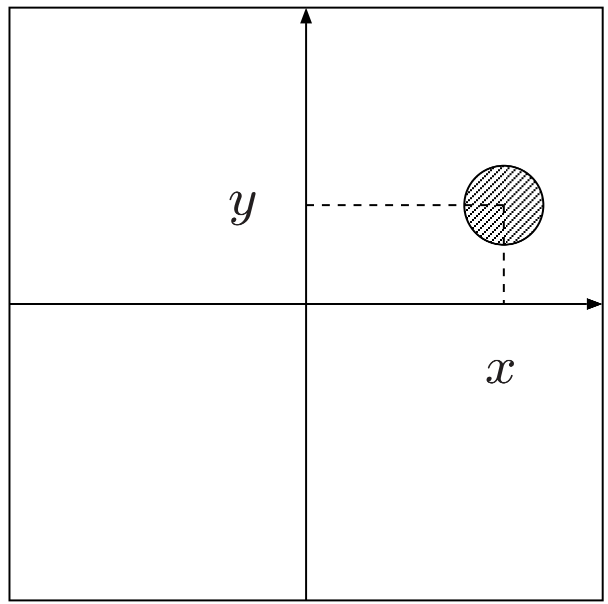
- Basic mathematical tools for control and image processing
- Tools for visual servo: cameras and software
- Image processing basics
- Nonlinear control and robot control
- Basic visual servo

What is visual servo



System

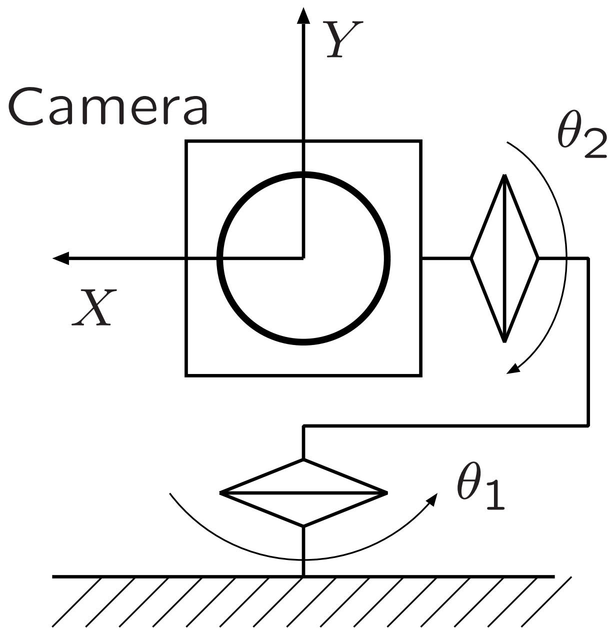


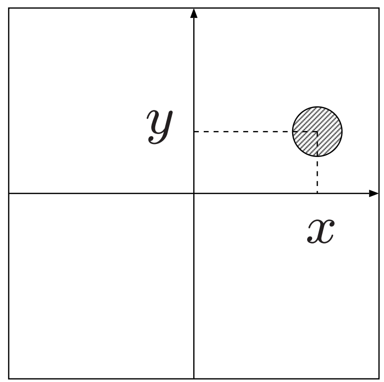


Control law

- Task: Keep object image $\mathbf{x} = [x, y]^\top$ at $\mathbf{x}^* = [0, 0]^\top$.
- Model: $\theta_1+ \to x+$, $\theta_2+ \to y+$
- Control law:

$$\dot{\boldsymbol{\theta}} = -\lambda(\mathbf{x} - \mathbf{x}^*) = -\lambda\mathbf{x}$$





Stability and sampling period

- Sampling period: T
- Motor angle during 1 period: $T\lambda\mathbf{x}$
- If λ is big, then the robot moves quickly.
- But if both λ and T are big, then the motor rotate too much and the response becomes vibratory.
- So when T is big we cannot increase λ.
- If delay exists, then the closed loop is easily become unstable.

Delay in the loop

- Image acquisition includes exposure, AD convert, data transfer from camera to PC, DMA transfer to CPU.
- At least 1 frame delay from exposure to CPU.
- At least 1 frame delay for image processing.
- NTSC has 1/15 sec delay.

Camera resolution

- Image intensity at (i, j) pixel: I_{ij}
- Center of mass

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{M_{00}} \sum_{i=1}^{N} \sum_{j=1}^{M} \left(I_{ij} \begin{bmatrix} i \\ j \end{bmatrix} \right)$$

$$M_{00} = \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij}$$

- In this case the resolution is not very critical.

Camera resolution

- On the other hand, for example template matching, find (d_x, d_y) that minimize

$$\epsilon(d_x, d_y) = \sum_{i=u-w}^{u+w} \sum_{j=v-w}^{v+w} (I(i,j) - J(i + d_x, j + d_y))^2$$

  where I is template and J is current image.
- With low resolution the features in the original scene is lost and local matching becomes useless.
- And it is fragile to digitization.

Pyramidal computation

- As a recipe to local matching pyramidal computation is useful.
- For example, matching with Gaussian low pass filter with different mask sizes are used (OpenCV cvGoodFeaturesTo-Track(), cvCalcOpticalFlowPyrLK()).

Feature matching

- For 3D reconstruction, feature matching with two images are needed.
- In this case, scale invariant matching methods e.g., SIFT, SURF, are used. SURF is included in OpenCV (from 1.1).

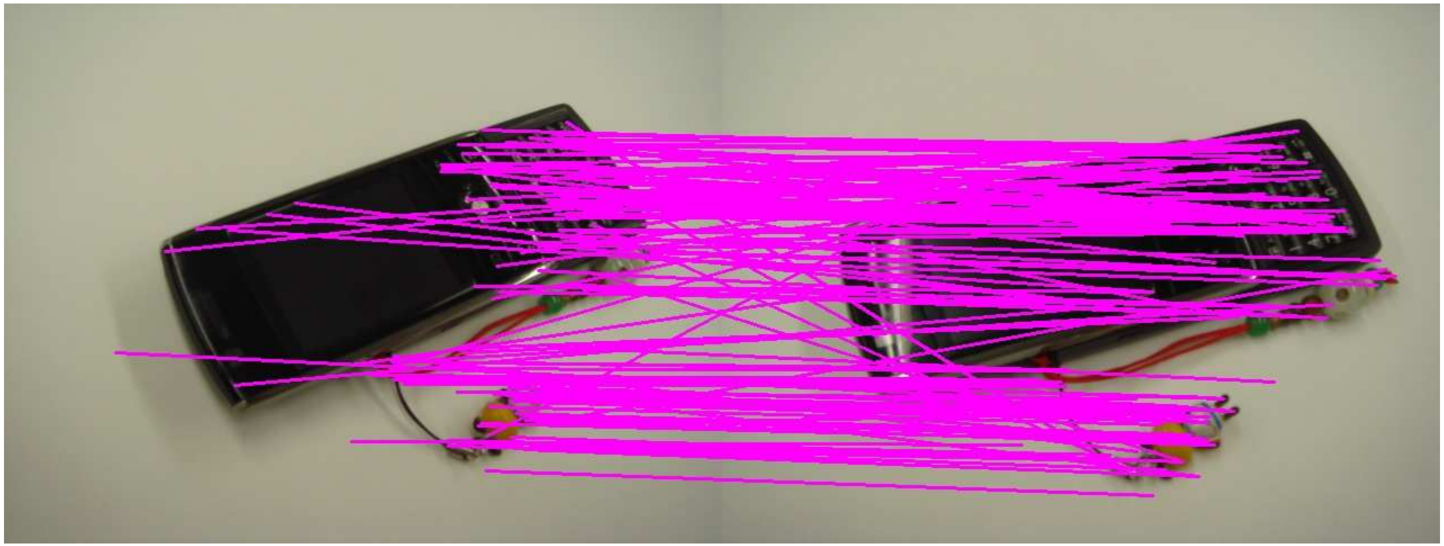
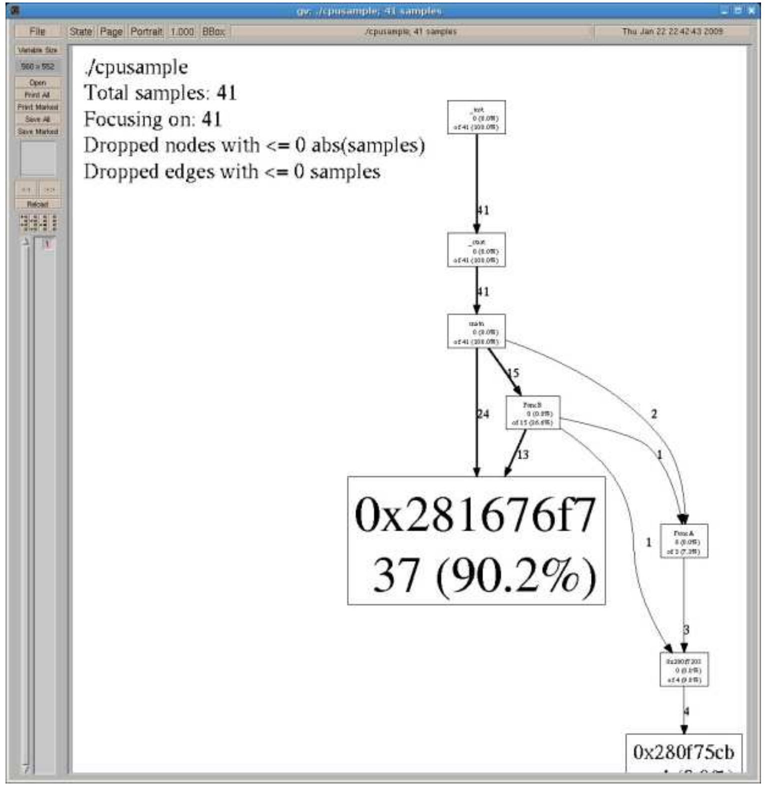


Image processing tools

- OpenCV is very handy and easy to use.
- Image processing uses linear algebra a lot so use of BLAS, LAPACK, ATLAS are effective to speed up.
- For Intel CPU, Intel Integrated Performance Primitives is useful.
- google-perftools: fast malloc, cpu profiler



Camera selection

ARTCAM-200MI (USB2.0 480Mbps)

Grasshopper™(IEEE1394B 800Mbps)

MC1364 EoSens® GE

(GigE Vision™1Gbps)

MC1362 EoSens® CL

(Camera Link™2.2Gbps)

Camera selection

	VGA	SXGA	UXGA
USB2.0	46	—	10
IEEE1394B	200	—	—
GigE Vision™	300	80	—
Camera Link™	1600	500	—

- VGA (640×480)
- SXGA (1280×1024)
- UXGA (1600×1200)

- ARTCAM-200MI (USB2.0 480Mbps)
- Dragonfly Express™(IEEE1394B 800Mbps)
- MC1364 EoSens® GE (GigE Vision™1Gbps)
- MC1362 EoSens® CL (Camera Link™2.2Gbps)

Interaction: image processing and control

Interaction: image processing and control

- To estimate the helicopter position, positions of markers are estimated.
- Set region of interest and extract features.
- If some markers are not found, these markers are occluded.
- If marker information from multiple cameras do not match, then do not use the camera that sent mismatch data.

References

- OpenCV: http://opencv.willowgarage.com/wiki/
- BLAS: http://www.netlib.org/blas/
- LAPACK: http://www.netlib.org/lapack/
- ATLAS: http://math-atlas.sourceforge.net/
- IPP: http://software.intel.com/en-us/intel-ipp/
- google-perftools: http://code.google.com/p/google-perftools/

Menu: Course I

- Basic mathematical tools for control and image processing
- Tools for visual servo: cameras and software
- Image processing basics
- Nonlinear control and robot control
- Basic visual servo

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

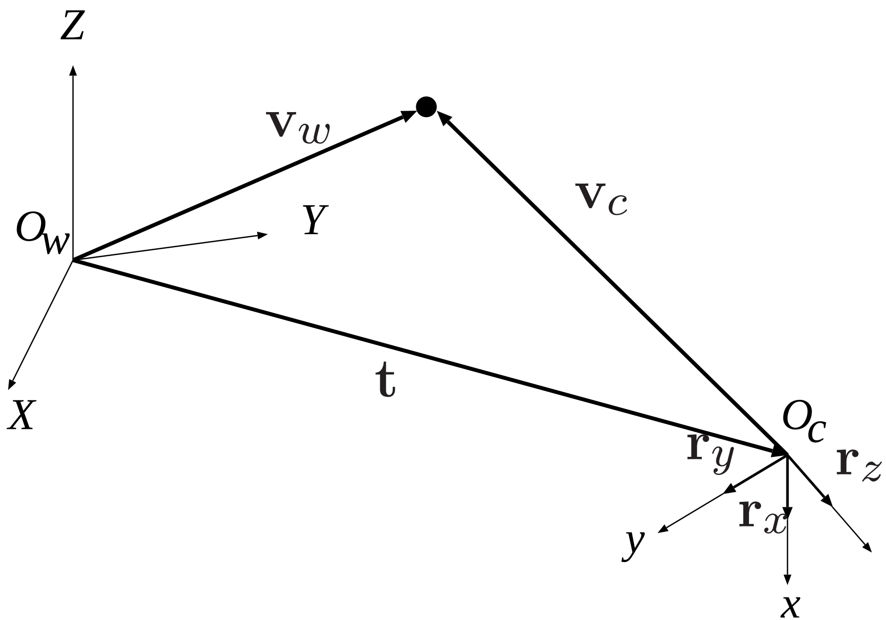
Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

Camera position and image

- Point at $\mathbf{v}_c = [x \;\; y \;\; z]^\top$ in camera coordinate system

$$\mathbf{v}_w = x\mathbf{r}_x + y\mathbf{r}_y + z\mathbf{r}_z + \mathbf{t} = \mathbf{R}\mathbf{v}_c + \mathbf{t}$$



Homogeneous matrix

- Augmented vector

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Homogeneous matrix of internal parameters

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Homogeneous transformation

- Rigid transformation (external parameters)

$$\mathbf{v}_w = x\mathbf{r}_x + y\mathbf{r}_y + z\mathbf{r}_z + \mathbf{t} = \mathbf{R}\mathbf{v}_c + \mathbf{t}$$

- Augment $\mathbf{v}_w$ and $\mathbf{v}_c$

$$\mathbf{v}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \quad \mathbf{v}_c = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

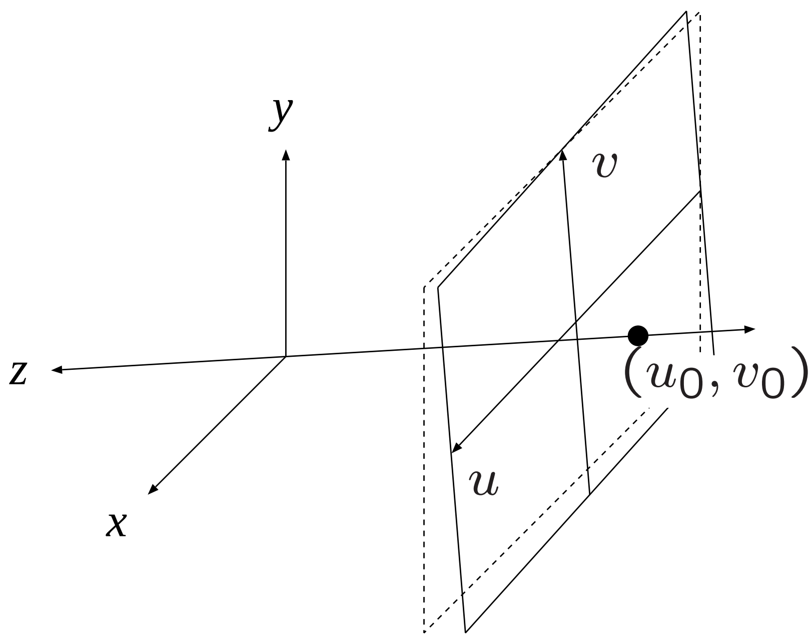
- Homogeneous transformation

$$\mathbf{v}_w = \mathbf{D}\mathbf{v}_c, \quad \mathbf{D} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}$$

Pixel coordinate system

- u, v are pixel coordinate

$$
\begin{aligned}
u &= k_u(x + y\cot\phi) + u_0 \\
v &= k_v \frac{y}{\sin\phi} + v_0
\end{aligned}
$$



Intrinsic parameters

- From position to pixels

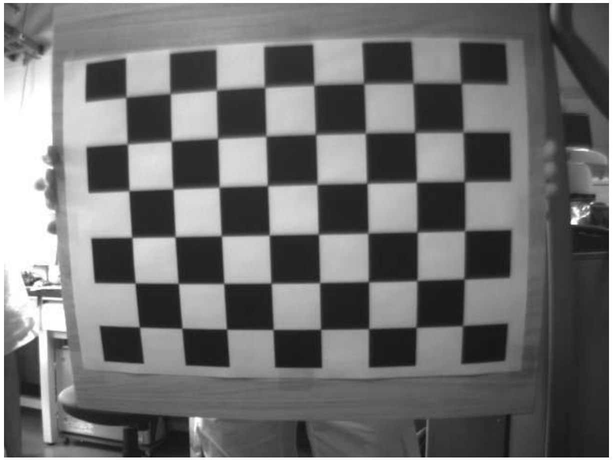
$$
\begin{aligned}
u &= k_u(x + y \cot\phi) + u_0 \\
v &= k_v \frac{y}{\sin\phi} + v_0
\end{aligned}
$$

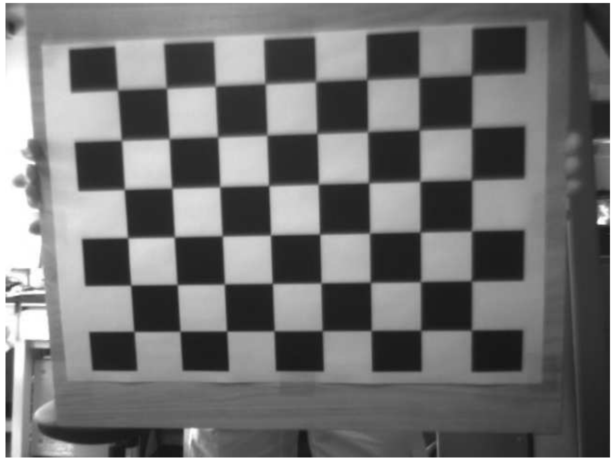
$$
s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}
$$

- Intrinsic parameter matrix

$$
\mathbf{A} = \begin{bmatrix} fk_u & fk_u \cot\phi & u_0 & 0 \\ 0 & fk_v / \sin\phi & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}
$$

Lens distortion





- Barrel distortion

$$
\begin{aligned}
u_d &= u(1 + k_1 r^2 + k_2 r^4) \\
v_d &= v(1 + k_1 r^2 + k_2 r^4)
\end{aligned}
$$

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

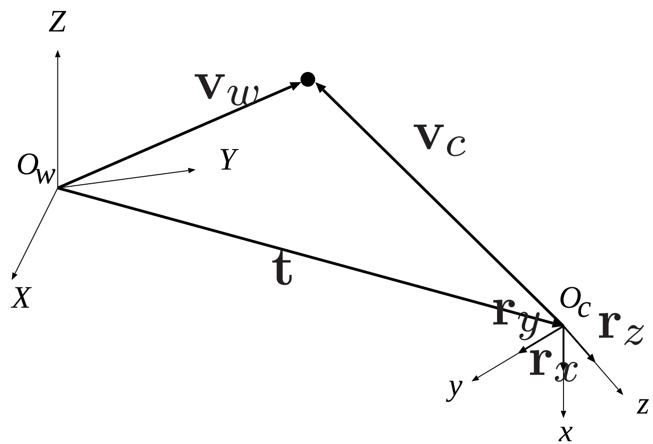
Camera calibration: setup

- Camera position

$$\mathbf{v}_w = \mathbf{R}\mathbf{v}_c + \mathbf{t}, \qquad \mathbf{v}_c = \mathbf{R}^\top \mathbf{v}_w - \mathbf{R}^\top \mathbf{t}$$

- Object image

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \mathbf{A}\,[\,\mathbf{R}^\top \quad -\mathbf{R}^\top \mathbf{t}\,] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



Calibration procedure

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{A}\,[\,\mathbf{R}^\top \quad -\mathbf{R}^\top\mathbf{t}\,]\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{P} = \mathbf{A}\,[\,\mathbf{R}^\top \quad -\mathbf{R}^\top\mathbf{t}\,]$$

- Put a checker board at a known position.
- Then $[X\ Y\ Z]^\top$ becomes a known vector.
- We have two equations per point.
- Solve for $\mathbf{P}$

Solve linear equations

- Let

$$\mathbf{P} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix}$$

- Then we have

$$\begin{aligned} u &= \frac{P_{11}X + P_{12}Y + P_{13}Z + P_{14}}{P_{31}X + P_{32}Y + P_{33}Z + P_{34}} \\ v &= \frac{P_{21}X + P_{22}Y + P_{23}Z + P_{24}}{P_{31}X + P_{32}Y + P_{33}Z + P_{34}} \end{aligned}$$

Calibration

- **Problem:** Find the solution $\mathbf{P} \in \mathbb{R}^{3\times 4}$ that minimizes

$$\sum_i \left\| s_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} - \mathbf{P} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \right\|^2 \to \text{min}$$

- **Algorithm:** For point i we have

$$\begin{aligned} s_i x_i &- (P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14}) \\ s_i y_i &- (P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24}) \\ s_i &- (P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}) \end{aligned}$$

and eliminate s_i.

Calibration algorithm

- Let $\mathbf{p} = [P_{11}, P_{12}, \ldots, P_{34}]^\top$, then

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \end{bmatrix} \mathbf{p} = \mathbf{q}_{ix}\mathbf{p}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \end{bmatrix} \mathbf{p} = \mathbf{q}_{iy}\mathbf{p}$$

- Assume $P_{34} = 1$, then we have.

$$\begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} \mathbf{q}_{1x} \\ \mathbf{q}_{1y} \\ \mathbf{q}_{2x} \\ \mathbf{q}_{2y} \\ \vdots \\ \mathbf{q}_{Ny} \end{bmatrix} \hat{\mathbf{p}}$$

Calibration algorithm

- Linear equations

$$\min_{\hat{\mathbf{p}}} \|\mathbf{y} - \mathbf{Q}\hat{\mathbf{p}}\|^2$$

where $\mathbf{y}$ and $\mathbf{Q}$ are the data vector and matrix.

- Since $\mathbf{Q}$ is a tall matrix, the solution is obtained by the generalized inverse.

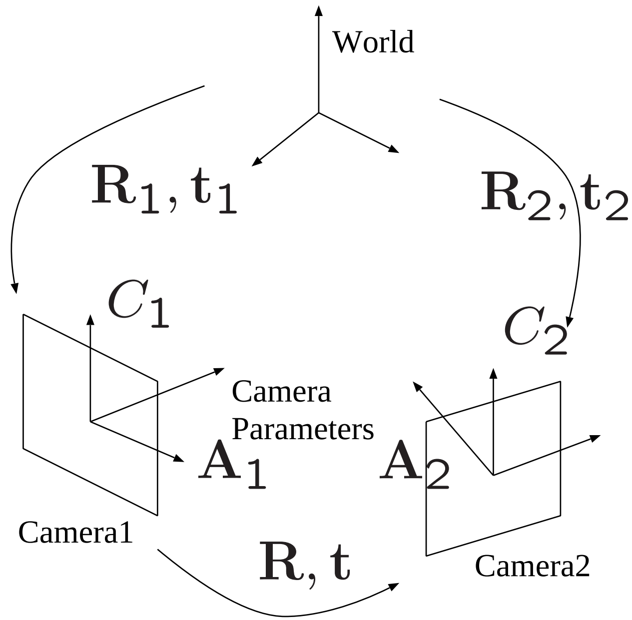
$$\hat{\mathbf{p}} = \mathbf{Q}^{\dagger}\mathbf{y}$$

- To find Lens distortion, nonlinear minimization is necessary. See the calibration package of OpenCV or MATLAB Image processing toolbox.

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

Stereo



- Two cameras look at the same point $\mathbf{v}_w = [X\ Y\ Z\ 1]^\top$.
- Images of that point in two cameras are

$$\mathbf{m}_1 = [u_1\ v_1\ 1]^\top, \quad \mathbf{m}_2 = [u_2\ v_2\ 1]^\top$$

Stereo

- Let $\mathbf{P}_1, \mathbf{P}_2$ be the projection matrix, then we have

$$\begin{aligned} s_1\mathbf{m}_1 &= \mathbf{P}_1\mathbf{v}_w \\ s_2\mathbf{m}_2 &= \mathbf{P}_2\mathbf{v}_w \end{aligned}$$

- Suppose $\mathbf{P}_1, \mathbf{P}_2$ are calibrated as

$$\begin{aligned} \mathbf{P}_1 &= \begin{bmatrix} P^1_{11} & P^1_{12} & P^1_{13} & P^1_{14} \\ P^1_{21} & P^1_{22} & P^1_{23} & P^1_{24} \\ P^1_{31} & P^1_{32} & P^1_{33} & P^1_{34} \end{bmatrix} \\ \mathbf{P}_2 &= \begin{bmatrix} P^2_{11} & P^2_{12} & P^2_{13} & P^2_{14} \\ P^2_{21} & P^2_{22} & P^2_{23} & P^2_{24} \\ P^2_{31} & P^2_{32} & P^2_{33} & P^2_{34} \end{bmatrix} \end{aligned}$$

Stereo

- Then we have

$$\begin{aligned} s_1 u_1 &= P^1_{11} X + P^1_{12} Y + P^1_{13} Z + P^1_{14} \\ s_1 v_1 &= P^1_{21} X + P^1_{22} Y + P^1_{23} Z + P^1_{24} \\ s_1 &= P^1_{31} X + P^1_{32} Y + P^1_{33} Z + P^1_{34} \end{aligned}$$

- Multiply u_1 and v_1 to the third equation

$$\begin{aligned} s_1 u_1 &= P^1_{31} u_1 X + P^1_{32} u_1 Y + P^1_{33} u_1 Z + P^1_{34} u_1 \\ s_1 v_1 &= P^1_{31} v_1 X + P^1_{32} v_1 Y + P^1_{33} v_1 Z + P^1_{34} v_1 \end{aligned}$$

Stereo

- Subtracting them from first and second equations, respectively, yields

$$\begin{bmatrix} P^1_{11} - P^1_{31}u_1 & P^1_{12} - P^1_{32}u_1 & P^1_{13} - P^1_{33}u_1 \\ P^1_{21} - P^1_{31}v_1 & P^1_{22} - P^1_{32}v_1 & P^1_{23} - P^1_{33}v_1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} P^1_{34}u_1 - P^1_{14} \\ P^1_{34}v_1 - P^1_{24} \end{bmatrix}$$

- Similarly we have

$$\begin{bmatrix} P^1_{11} - P^1_{31}u_1 & P^1_{12} - P^1_{32}u_1 & P^1_{13} - P^1_{33}u_1 \\ P^1_{21} - P^1_{31}v_1 & P^1_{22} - P^1_{32}v_1 & P^1_{23} - P^1_{33}v_1 \\ P^2_{11} - P^2_{31}u_2 & P^2_{12} - P^2_{32}u_2 & P^2_{13} - P^2_{33}u_2 \\ P^2_{21} - P^2_{31}v_2 & P^2_{22} - P^2_{32}v_2 & P^2_{23} - P^2_{33}v_2 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} P^1_{34}u_1 - P^1_{14} \\ P^1_{34}v_1 - P^1_{24} \\ P^2_{34}u_2 - P^2_{14} \\ P^2_{34}v_2 - P^2_{24} \end{bmatrix}$$

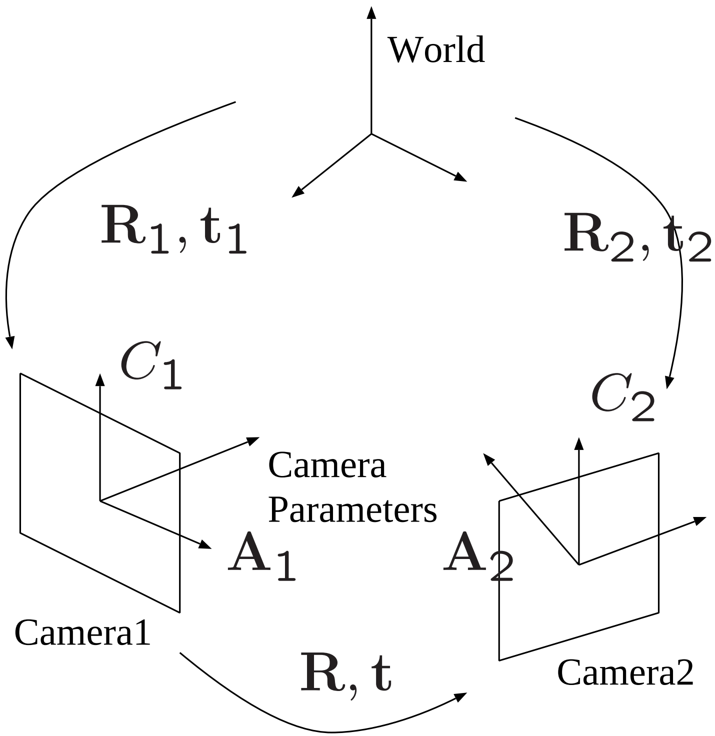
i.e.,

$$\mathbf{M}\mathbf{v}_w = \mathbf{b} \quad \Longrightarrow \quad \mathbf{v}_w = \mathbf{M}^\dagger \mathbf{b}$$

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

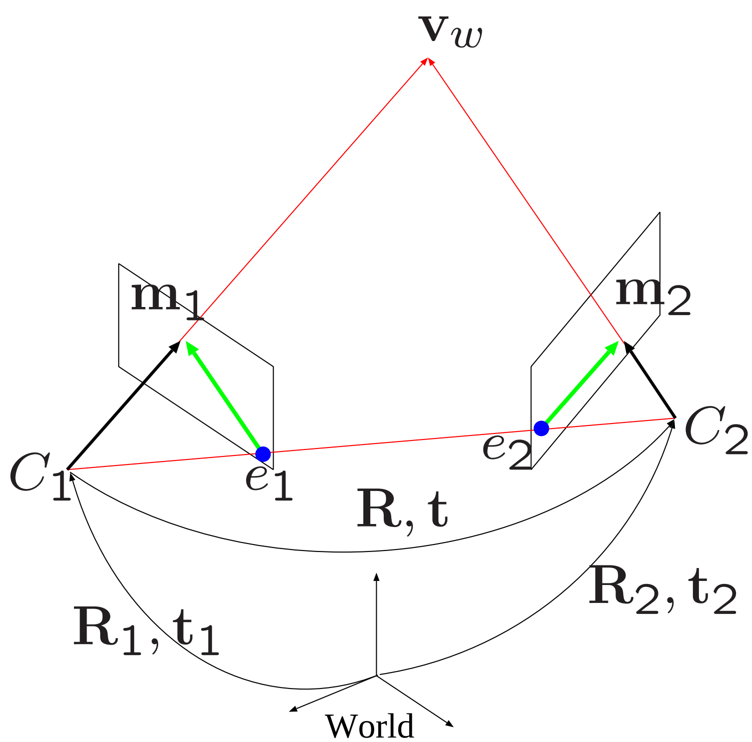
Two cameras setup



- Intrinsic parameters: $\mathbf{A}_1, \mathbf{A}_2$
- Geometrical relationship

$$\mathbf{R} = \mathbf{R}_1^\top \mathbf{R}_2, \quad \mathbf{t} = \mathbf{R}_1^\top (\mathbf{t}_2 - \mathbf{t}_1)$$

Epipolar geometry



$$
\begin{aligned}
s_1 \mathbf{m}_1 &= \mathbf{A}_1 \left[\mathbf{R}_1^\top \quad -\mathbf{R}_1^\top \mathbf{t}_1 \right] \mathbf{v}_w \\
s_2 \mathbf{m}_2 &= \mathbf{A}_2 \left[\mathbf{R}_2^\top \quad -\mathbf{R}_2^\top \mathbf{t}_2 \right] \mathbf{v}_w
\end{aligned}
$$

Epipolar geometry

$$\begin{aligned} s_1\mathbf{m}_1 &= \mathbf{A}_1\,[\,\mathbf{R}_1^\top \quad -\mathbf{R}_1^\top\mathbf{t}_1\,]\,\mathbf{v}_w \\ s_2\mathbf{m}_2 &= \mathbf{A}_2\,[\,\mathbf{R}_2^\top \quad -\mathbf{R}_2^\top\mathbf{t}_2\,]\,\mathbf{v}_w \end{aligned}$$

$$\begin{aligned} s_1(\mathbf{A}_1\mathbf{R}_1^\top)^{-1}\mathbf{m}_1 &= \mathbf{v}_w - \mathbf{t}_1 \\ s_2(\mathbf{A}_2\mathbf{R}_2^\top)^{-1}\mathbf{m}_2 &= \mathbf{v}_w - \mathbf{t}_2 \end{aligned}$$

$$s_1\mathbf{R}_1\mathbf{A}_1^{-1}\mathbf{m}_1 - s_2\mathbf{R}_2\mathbf{A}_2^{-1}\mathbf{m}_2 = \mathbf{t}_2 - \mathbf{t}_1$$

$$s_1\mathbf{A}_1^{-1}\mathbf{m}_1 - s_2\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2 = \mathbf{t}$$

- Interpretation: Three vectors $\mathbf{A}_1^{-1}\mathbf{m}_1$, $\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2$, $\mathbf{t}$ are on the same plane.

Epipolar geometry

$$s_1\mathbf{A}_1^{-1}\mathbf{m}_1 - s_2\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2 = \mathbf{t}$$

- Interpretation: Three vectors $\mathbf{A}_1^{-1}\mathbf{m}_1$, $\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2$, $\mathbf{t}$ are on the same plane.

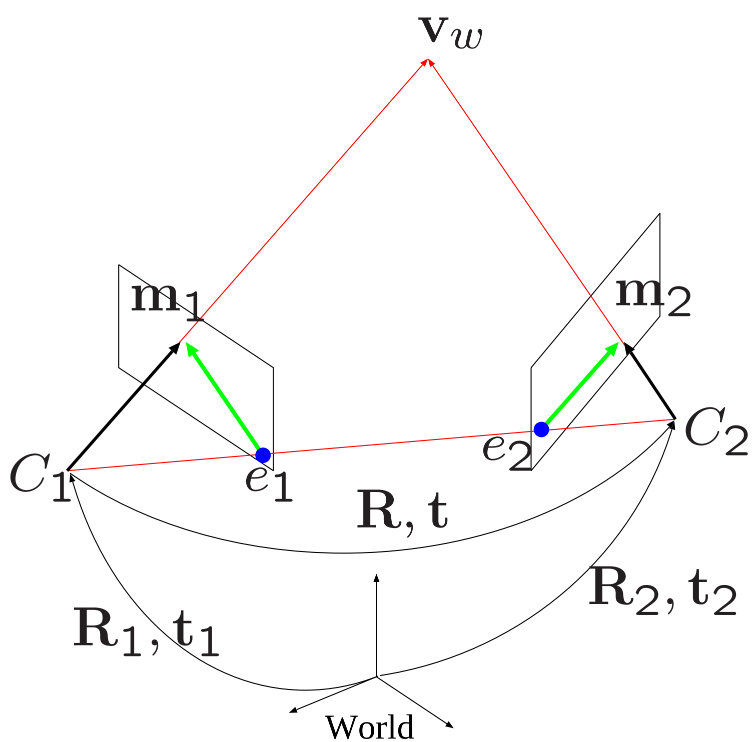


Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

Skew-symmetric Matrix

- Let vector cross product operator be $\wedge$.
- $\mathbf{A}_1^{-1}\mathbf{m}_1$ and $\mathbf{t} \wedge (\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2)$ are orthogonal.
- Introduce a skew symmetric matrix of $\mathbf{t} = [t_x \ t_y \ t_z]^\top$

$$
\begin{aligned}
[\mathbf{t}]_\wedge &= \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \\
[\mathbf{t}]_\wedge \mathbf{v} &= \begin{bmatrix} t_y v_z - t_z v_y \\ t_z v_x - t_x v_z \\ t_x v_y - t_y v_x \end{bmatrix} = \mathbf{t} \wedge \mathbf{v}
\end{aligned}
$$

Fundamental Matrix

- Since

$$s_1\mathbf{A}_1^{-1}\mathbf{m}_1 - s_2\mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2 = \mathbf{t}$$

we have

$$[\mathbf{t}]_\wedge \mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2 \perp \mathbf{A}_1^{-1}\mathbf{m}_1$$

and

$$\mathbf{m}_1^\top (\mathbf{A}_1^{-1})^\top [\mathbf{t}]_\wedge \mathbf{R}\mathbf{A}_2^{-1}\mathbf{m}_2 = 0$$

- Fundamental Matrix

$$\mathbf{F} = (\mathbf{A}_1^{-1})^\top [\mathbf{t}]_\wedge \mathbf{R}\mathbf{A}_2^{-1}$$

- Fundamental Equation

$$\mathbf{m}_1^\top \mathbf{F}\mathbf{m}_2 = 0$$

Essential Matrix

- Fundamental Equation

$$\mathbf{m}_1^\top \mathbf{F} \mathbf{m}_2 = 0$$

$$\mathbf{F} = (\mathbf{A}_1^{-1})^\top [\mathbf{t}]_\wedge \mathbf{R} \mathbf{A}_2^{-1}$$

- Essential Matrix

$$\mathbf{E} = [\mathbf{t}]_\wedge \mathbf{R}$$

- Essential Equation

$$\mathbf{v}_1^\top \mathbf{E} \mathbf{v}_2 = 0$$

where

$$\mathbf{v}_1 = \mathbf{A}_1^{-1} \mathbf{m}_1, \quad \mathbf{v}_2 = \mathbf{A}_2^{-1} \mathbf{m}_2$$

Property of essential matrix

- For any rotation matrix $\mathbf{R}$ and any skew symmetric matrix $\mathbf{T}$ define a set of essential matrix by

$$\mathbb{E} = \{\mathbf{E} : \mathbf{E} = \mathbf{TR}, \quad \mathbf{R}^\top \mathbf{R} = \mathbf{I} \text{ and } \mathbf{T} = [\mathbf{t}]_\wedge\}$$

- For any matrix $\mathbf{Q} \in \mathbb{E}$ is an essential matrix.
- And all essential matrix have eigenvalues $\lambda, \lambda, 0$ (two duplicated and one zero).

$$\mathbf{Q} \in \mathbb{E} \quad \Longleftrightarrow \quad \mathbf{Q} = \mathbf{U\Sigma V}^\top \text{ and } \mathbf{\Sigma} = \text{diag}\{\lambda, \lambda, 0\}$$

Proof of the property

$(\Longrightarrow)$: Let $\mathbf{t} = \|t\|\mathbf{z}$ and SVD $\mathbf{Q}$.

- Since

$$\mathbf{Q}\mathbf{Q}^\top = \mathbf{T}\mathbf{R}\mathbf{R}^\top\mathbf{T}^\top = \mathbf{T}\mathbf{T}^\top = -\mathbf{T}^2$$

- Multiply a orthonormal basis $[\mathbf{x}\ \mathbf{y}\ \mathbf{z}]$ to $\mathbf{T}^2$

$$\begin{aligned}
\mathbf{T}^2[\mathbf{x}\ \mathbf{y}\ \mathbf{z}] &= \mathbf{t} \wedge \mathbf{t} \wedge [\mathbf{x}\ \mathbf{y}\ \mathbf{z}] = \|t\|^2[-\mathbf{x}\ \ -\mathbf{y}\ \mathbf{0}] \\
&= [\mathbf{x}\ \mathbf{y}\ \mathbf{z}]\mathrm{diag}\{-\|t\|^2, -\|t\|^2, 0\} \\
\mathbf{T}^2 &= [\mathbf{x}\ \mathbf{y}\ \mathbf{z}]\mathrm{diag}\{-\|t\|^2, -\|t\|^2, 0\}[\mathbf{x}\ \mathbf{y}\ \mathbf{z}]^\top
\end{aligned}$$

- Thus

$$\begin{aligned}
\mathbf{Q}\mathbf{Q}^\top &= [\mathbf{x}\ \mathbf{y}\ \mathbf{z}]\mathrm{diag}\{\|t\|^2, \|t\|^2, 0\}[\mathbf{x}\ \mathbf{y}\ \mathbf{z}]^\top \\
\mathbf{Q} &= [\mathbf{x}\ \mathbf{y}\ \mathbf{z}]\mathrm{diag}\{\|t\|, \|t\|, 0\}\mathbf{V}^\top = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top
\end{aligned}$$

where $\mathbf{V}$ is any orthogonal matrix.

Proof of the property

$(\Longleftarrow)$: Let $\mathbf{Q} = \mathbf{U}\mathbf{\Sigma_0}\mathbf{V}^\top$ where $\mathbf{\Sigma_0} = \text{diag}\{\lambda_0, \lambda_0, 0\}$.

- Define $\mathbf{R}_z$ and decompose $\mathbf{Q}$

$$
\begin{aligned}
\mathbf{R}_z &= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
\mathbf{Q} &= \mathbf{U}\mathbf{\Sigma}_0\mathbf{R}_z^\top\mathbf{U}^\top\mathbf{U}\mathbf{R}_z\mathbf{V}^\top = \mathbf{T}_0\mathbf{R}_0 \\
\mathbf{T}_0 &= \mathbf{U}\mathbf{\Sigma}_0\mathbf{R}_z^\top\mathbf{U}^\top, \quad \mathbf{R}_0 = \mathbf{U}\mathbf{R}_z\mathbf{V}^\top
\end{aligned}
$$

- Show that $\mathbf{R}_0$ is rotation matrix and $\mathbf{T}_0$ is skew-symmetric.

$$
\mathbf{R}_z\mathbf{\Sigma}_0 = \mathbf{\Sigma}_0\mathbf{R}_z = -\mathbf{\Sigma}_0\mathbf{R}_z^\top
$$

$$
\begin{aligned}
\mathbf{R}_0^\top\mathbf{R}_0 &= \mathbf{V}\mathbf{R}_z^\top\mathbf{U}^\top\mathbf{U}\mathbf{R}_z\mathbf{V}^\top = \mathbf{I} \\
\mathbf{T}_0^\top &= \mathbf{U}\mathbf{R}_z\mathbf{\Sigma}_0\mathbf{U}^\top = \mathbf{U}\mathbf{\Sigma}_0\mathbf{R}_z\mathbf{U}^\top = -\mathbf{U}\mathbf{\Sigma}_0\mathbf{R}_z^\top\mathbf{U}^\top = -\mathbf{T}_0
\end{aligned}
$$

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

Eight point algorithm

- When a corresponding point from two cameras is found then we have one essential equation.
- For N points, we have

$$\begin{aligned}\mathbf{m}_{1j}^\top \mathbf{E} \mathbf{m}_{2j} &= 0 \quad \text{for} \quad j = 1, \ldots N \\ \mathbf{E} &= \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}\end{aligned}$$

  where $\mathbf{m}_{ij}$ is the image coordinate of j-th point in i-th camera.
- $\mathbf{E}$ has 9 elements but it has 1 free dof because we can multiply a scalar for essential equation.
- Thus 8 points are sufficient to estimate $\mathbf{E}$.

Eight point algorithm

$$
\begin{aligned}
\mathbf{Be} &= \mathbf{0}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_N \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} e_{11} \\ e_{12} \\ \vdots \\ e_{33} \end{bmatrix}, \\
\mathbf{b}_i &= [u_{2j}u_{1j} \quad u_{2j}v_{1j} \quad u_{2j}s_{1j} \\
&\qquad v_{2j}u_{1j} \quad v_{2j}v_{1j} \quad v_{2j}s_{1j} \\
&\qquad\quad s_{2j}u_{1j} \quad s_{2j}v_{1j} \quad s_{2j}s_{1j}] \\
\|\mathbf{e}\| &= 1
\end{aligned}
$$

Problem: Under $\|\mathbf{e}\| = 1$, find $\mathbf{e}$ that minimizes $\|\mathbf{Be}\|$.

Solution: SVD $\mathbf{B}$. $\mathbf{e}$ is the smallest singular vector. After that, construct $\mathbf{E}$ from $\mathbf{e}$ and modify it so that it has singular values $\lambda, \lambda, 0$.

Finding motion parameters

- SVD of $\mathbf{E}$

$$\mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

- Then $\mathbf{t}$ and $\mathbf{R}$ are given by

$$\mathbf{t} = \sigma_1 \mathbf{u}_3, \quad \mathbf{R} = \mathbf{U}\mathbf{R}_z\mathbf{V}^\top$$

where $\mathbf{u}_3$ is the third column of $\mathbf{U}$.

Code

1. Construct $\mathbf{B}$ from $\{(\mathbf{m}_{1i}, \mathbf{m}_{2i}), i = 1, \ldots, N\}$

```
B=zeros(N,9);
for i=1:N
  u1=m1(1,i);  v1=m1(2,i);  w1=m1(3,i);
  u2=m2(1,i);  v2=m2(2,i);  w2=m2(3,i);
  B(i,:)=[u2*u1, u2*v1, u2*w1,
          v2*u1, v2*v1, v2*w1,
          w2*u1, w2*v1, w2*w1];
end
```

2. Find $\mathbf{e}$ ($\|\mathbf{e}\| = 1$) such that $\|\mathbf{Be}\| \to$ min.

```
[U D V]=svd(B);  e=V(:,9);
```

3. Construct $\mathbf{E}$ from $\mathbf{e}$, where trace$\mathbf{E}^\top\mathbf{E} = 2$ [Hartley].

```
E=sqrt(2)*[e(1:3)'; e(4:6)'; e(7:9)'];
# residual=trace(m2'*E*m1);
```

4. Find $\hat{\mathbf{E}} = \mathbf{TR}$ that minimize $\|\hat{\mathbf{E}} - \mathbf{E}\|$.

```
[U D V]=svd(E);
D=diag((D(1)+D(2))/2, (D(1)+D(2))/2, 0);
hatE=U*D*V';
```

5. Decompose $\hat{\mathbf{E}}$ to find $\mathbf{R}$ and $\mathbf{T}$.

```
[U, D, V]=svd(hatE);
t=U(:,3);
Rz=[0 1 0; -1 0 0;0 0 1];
R1=U*Rz*V';
R2=U*Rz'*V';
```

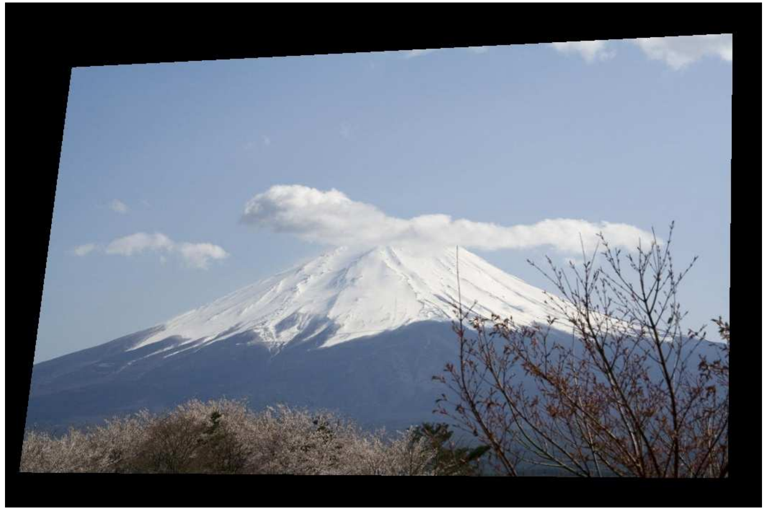
6. Select a feasible $\mathbf{R}$ from `R1`, `R2`.

Image Processing

- Camera model
- Camera calibration
- Stereo (3D estimation basics)
- Epipolar geometry
- Fundamental matrix, Essential matrix
- Eight point algorithm (3D estimation without calibration)
- Homography

Homography matrix

Show a picture from different viewpoint.



Homography matrix

- Suppose that all points are on an plane.
- A complete correspondence of two images from different view points is obtained by a **Homography matrix** $\mathbf{H}$.
- Points in camera 1 and 2 are give by

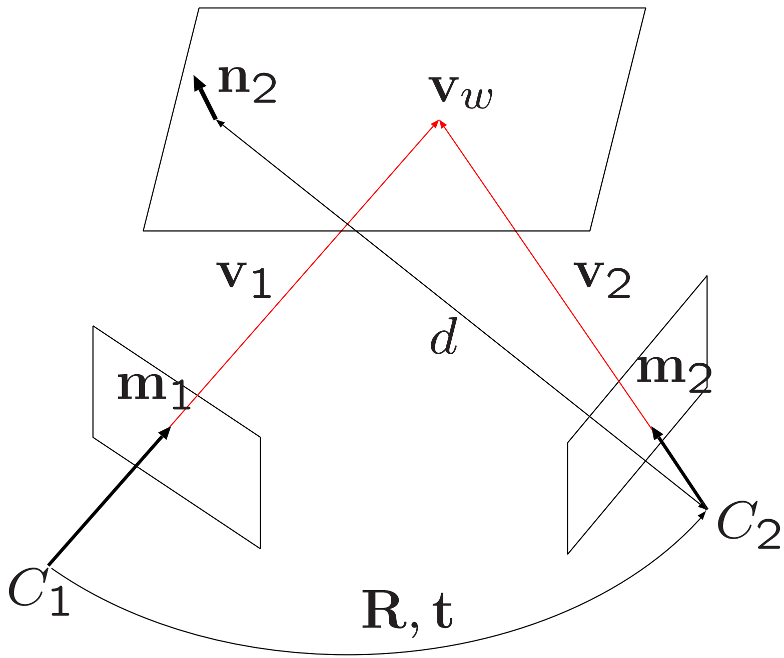
$$\mathbf{m}_1 = [u_1 \ v_1 \ 1]^\top, \quad \mathbf{m}_2 = [u_2 \ v_2 \ 1]^\top$$

  then we have

$$s\mathbf{m}_1 = \mathbf{H}\mathbf{m}_2$$

- This equation holds for all points on a plane.

Homography matrix



- $\mathbf{n}_2$ is a normal vector of the plane.
- $\mathbf{n}_2, \mathbf{v}_2, \mathbf{m}_2$ are expressed in camera 2 coordinate system, others are expressed in camera 1 coordinate system.

Homography matrix

- d_2 is distance from the point C_2 and the plane

$$\mathbf{n}_2^\top \mathbf{v}_2 = d_2$$

- $\mathbf{R}$ and $\mathbf{t}$ are transformation from camera 2 to 1:

$$\mathbf{v}_1 = \mathbf{R}\mathbf{v}_2 + \mathbf{t}$$

- Since

$$\frac{\mathbf{n}_2^\top}{d_2}\mathbf{v}_2 = 1$$

  we have the following equation by multiplying $\mathbf{t}$ from left.

$$\frac{\mathbf{t}\mathbf{n}_2^\top}{d_2}\mathbf{v}_2 = \mathbf{t}$$

- Finally we obtain

$$\mathbf{v}_1 = \left(\mathbf{R} + \frac{\mathbf{t}\mathbf{n}_2^\top}{d_2}\right)\mathbf{v}_2$$

Homography matrix

- Image of $\mathbf{v}_w$ from cameras 1 and 2: $\mathbf{m}_1, \mathbf{m}_2$

$$s_1\mathbf{m}_1 = \mathbf{A}_1\mathbf{v}_1, \quad s_2\mathbf{m}_2 = \mathbf{A}_2\mathbf{v}_2$$

- Thus we have

$$s\mathbf{m}_1 = \mathbf{H}\mathbf{m}_2$$

where

$$\mathbf{H} = \mathbf{A}_1^{-1}\left(\mathbf{R} + \frac{\mathbf{t}\mathbf{n}_2^\top}{d_2}\right)\mathbf{A}_2$$

Points to homography matrix

- Homography equation

$$s\begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} u_2 & v_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & u_2 & v_2 & 1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ \vdots \\ h_{33} \end{bmatrix}$$

- Substitute $s = h_{31}u_2 + h_{32}v_2 + h_{33}$

$$\begin{bmatrix} u_2 & v_2 & 1 & 0 & 0 & 0 & -u_1u_2 & -u_1v_2 & -u_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -v_1u_2 & -v_1v_2 & -v_1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ \vdots \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Points to homography matrix

- **Problem:** Find $\mathbf{h}$ that satisfy

$$\begin{bmatrix} u_2 & v_2 & 1 & 0 & 0 & 0 & -u_1u_2 & -u_1v_2 & -u_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -v_1u_2 & -v_1v_2 & -v_1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ \vdots \\ h_{33} \end{bmatrix} \to \text{min}$$

- **Solution:** Using 4 points on a plane, the solution can be obtained using SVD (under $\|\mathbf{h}\| = 1$).

Code

1. Construct data matrix

```
C=zeros(2*m,9);
for i = 1:m
  C(2*(i-1)+1:2*i,:)...
  =[u2(i) v2(i) 1 0 0 0 -u1(i)*u2(i) -u1(i)*v2(i) -u1(i);
    0 0 0 u2(i) v2(i) 1 -v1(i)*u2(i) -v1(i)*v2(i) -v1(i)];
end
```

2. SVD

```
[U,S,V]=svd(C);
```

3. Resize

```
sol = V(:,9);
H = [sol(1:3,1)';sol(4:6,1)';sol(7:9,1)'];
```

Position estimation from Homography

- When $\mathbf{A}_1, \mathbf{A}_2$ are known, we can compute $\mathbf{R}, \mathbf{t}, \mathbf{n}, d$ based on

$$\mathbf{H} = \mathbf{A}_1^{-1} \left(\mathbf{R} + \frac{\mathbf{t}\mathbf{n}_2^\top}{d_2} \right) \mathbf{A}_2$$

while a scale indefiniteness of $\mathbf{t}$ remains.

- First using intrinsic parameters we have

$$\hat{\mathbf{H}} = \mathbf{R} + \frac{\mathbf{t}\mathbf{n}_2^\top}{d_2}$$

Position estimation from Homography

- And SVD

$$\hat{\mathbf{H}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

where

$$\boldsymbol{\Sigma} = d'\mathbf{R}' + \mathbf{t}'\mathbf{n}'^\top$$

and

$$\begin{aligned} \mathbf{R} &= s\mathbf{U}\mathbf{R}'\mathbf{V}^\top, \quad \mathbf{t} = \mathbf{U}\mathbf{t}', \quad \mathbf{n}_2 = \mathbf{V}\mathbf{n}', \\ d_2 &= sd', \quad s = \mathrm{det}(\mathbf{U})\mathrm{det}(\mathbf{V}) \end{aligned}$$

Position estimation from Homography

- Let $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ be a orthonormal basis and let

$$\mathbf{n}' = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + x_3\mathbf{e}_3, \qquad \sum_{i=1}^{3} x_i^2 = 1$$

- Multiply $\mathbf{e}_i$ to

$$\mathbf{\Sigma} = d'\mathbf{R}' + \mathbf{t}'\mathbf{n}'^{\top}$$

yields

$$\sigma_i \mathbf{e}_i = d'\mathbf{R}'\mathbf{e}_i + \mathbf{t}x_i \quad \text{for} \quad i = 1, 2, 3$$

and

$$d'\mathbf{R}'(x_j\mathbf{e}_i - x_i\mathbf{e}_j) = \sigma_i x_j \mathbf{e}_i - \sigma_j x_i \mathbf{e}_j \quad \text{for} \quad i \neq j$$

Position estimation from Homography

- Since $\mathbf{R}$ preserve vector norm, we have

$$\begin{aligned}(d'^2 - \sigma_2^2)x_1^2 + (d'^2 - d_1^2)x_2^2 &= 0\\(d'^2 - \sigma_3^2)x_2^2 + (d'^2 - d_2^2)x_3^2 &= 0\\(d'^2 - \sigma_1^2)x_3^2 + (d'^2 - d_3^2)x_1^2 &= 0\end{aligned}$$

- Viewing these equations as a set of linear equations for x_1^2, x_2^2, x_3^2, then the determinant of the coefficient matrix is zero.

$$(d'^2 - \sigma_1^2)(d'^2 - \sigma_2^2)(d'^2 - \sigma_3^2) = 0$$

Position estimation from Homography

- Classify by the number of duplication of singular values $\sigma_1, \sigma_2, \sigma_3$ of $\hat{\mathbf{H}}$
- All singular values are different ($\sigma_1 > \sigma_2 > \sigma_3$)
  - $d' = \sigma_1$ or $d' = \sigma_3$ are impossible. Because if $d' = \sigma_1$ then we have $(\sigma_1^2 - \sigma_3^2)x_2^2 + (\sigma^2 - d_2^2)x_3^2 = 0$ and finally $x_1 = x_2 = x_3 = 0$. $d' = \sigma_3$ is also excluded.
  - Since $d' = \pm\sigma_2$, we have

$$
\begin{aligned}
x_1 &= \sqrt{\frac{\sigma_1^2 - \sigma_2^2}{\sigma_1^2 - \sigma_3^2}} \\
x_2 &= 0 \qquad\qquad\qquad\qquad \epsilon_1, \epsilon_2 = \pm 1 \\
x_3 &= \sqrt{\frac{\sigma_2^2 - \sigma_3^2}{\sigma_1^2 - \sigma_3^2}}
\end{aligned}
$$

- Assume $d' > 0$. Case for $d' < 0$ is similar.
- Since $d' = d_2, x_2 = 0$, $\mathbf{e}_2 = \mathbf{R}'\mathbf{e}_2$ and

$$\mathbf{R}' = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

- Thus

$$\begin{aligned} \sin\theta &= (\sigma_1 - \sigma_3)\frac{x_1 x_3}{\sigma_2} = \epsilon_1\epsilon_2\frac{\sqrt{(\sigma_1^2 - \sigma_2^2)(\sigma_2^2 - \sigma_3^2)}}{(\sigma_1 + \sigma_3)\sigma_2} \\ \cos\theta &= \frac{\sigma_1 x_3^2 + \sigma_3 x_1^2}{\sigma_2} = \frac{\sigma_2^2 + \sigma_1\sigma_3}{(\sigma_1 + \sigma_3)\sigma_2} \end{aligned}$$

- And finally

$$\mathbf{t} = (d_1 - d_3)\begin{bmatrix} x_1 \\ 0 \\ -x_3 \end{bmatrix}$$

- We have 4 cases due to the sign of ϵ_1, ϵ_2.

- One duplicate singular values ($\sigma_1 = \sigma_2 > \sigma_3$ of $\sigma_1 > \sigma_2 = \sigma_1$)
  - Let $d' = \sigma_1 = \sigma_2$. Case for $\sigma_2 = \sigma_3$ is similar.
  - We have $x_1 = x_2 = 0, x_3 = \epsilon_1 = \pm 1$ and

$$\mathbf{R}' = \mathbf{I}, \quad \mathbf{t} = (d_3 - d_1)\mathbf{n}'$$

- All singular values are duplicated ($\sigma_1 = \sigma_2 = \sigma_3$).
  - $d' = d_1 = d_2 = d_3$. We cannot find x_1, x_2, x_3 and thus

$$\mathbf{R}' = \mathbf{I}, \quad \mathbf{t} = \mathbf{0}$$

Code

1. SVD

```
[U,D,V]=svd(H);
d1=D(1,1);  d2=D(2,2);  d3=D(3,3);
suv=det(U)*det(V);      d=d2;
```

2. No duplicate singular values (compute for 4 cases). `n0` is a normal vector of the plane

```
n0=[0;0;-1];
x1=sqrt((d1*d1-d2*d2)/(d1*d1-d3*d3));
x3=-sqrt((d2*d2-d3*d3)/(d1*d1-d3*d3));
n=[x1; 0; x3];          t=(d1-d3)*[x1;0;-x3];
st=(d1-d3)*x1*x3/d2;  ct=(d1*x3*x3+d3*x1*x1)/d2;
R=[ct 0 -st;0 1 0;st 0 ct];
```

```
R1=suv*U*R*V';  t1=U*t;  n1=V*n;  n00=R*n0;
if (n00(3)<0)
    yn(1)=norm(n0-n1);
else
    yn(1)=10000;
end
```

3. Selection

```
[minn,index]=min(yn);
switch index
    case 1
        R=R1;  t=t1;  n=n1;  d=suv*d;
    case 2
        :  % similar
end
H=R+t/d*n0';
```

Menu: Course I

- Basic mathematical tools for control and image processing
- Tools for visual servo: cameras and software
- Image processing basics
- Nonlinear control and robot control
- Basic visual servo

State equation

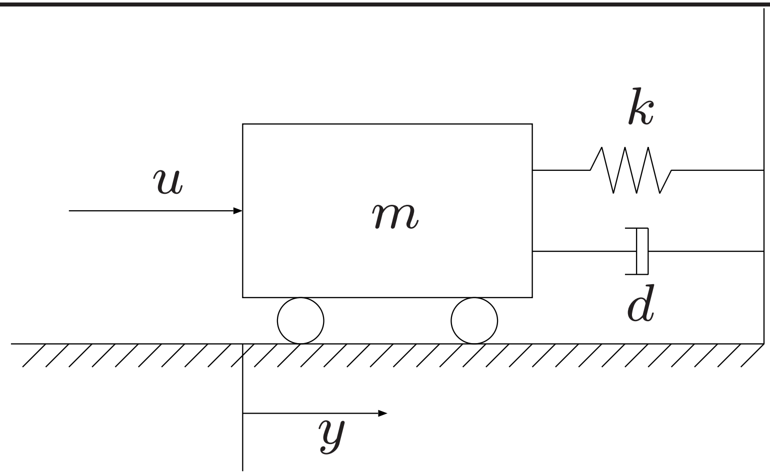
- State: $\mathbf{x}(t)$
- Input: $\mathbf{u}(t)$
- State equation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

- Measurement: $\mathbf{y}(t)$

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t))$$

Spring, Mass, Damper system



- Dynamical equation

$$m\ddot{y} + d\dot{y} + ky = u$$

- State

$$x_1 = y, \quad x_2 = \dot{y}$$

- State equation

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -d/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u$$

Linear System

- State equation

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)\\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t)\end{aligned}$$

- **Equilibrium point**: When the input is zero then the state will remain at this point.

$$\mathbf{x}(t) = \mathbf{x}_e \quad \text{and} \quad \dot{\mathbf{x}}(t) = 0 \quad \text{when} \quad \mathbf{u}(t) = 0$$

Stability

- Autonomous system

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$$

1. $\mathbf{x}(t) = 0$ is the (only one) equilibrium point.
2. For initial state $\mathbf{x}(0) = \mathbf{x}_0$, the state will follow

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}_0$$

where

$$e^{\mathbf{A}t} = \mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2 t^2}{2!} + \frac{\mathbf{A}^3 t^3}{3!} + \cdots = \sum_{i=0}^{\infty} \frac{\mathbf{A}^i t^i}{i!}$$

Stability

- By diagonalizing the matrix, we have $e^{\lambda_i t}$ at the diagonal element.

$$\mathbf{T}e^{\mathbf{A}t}\mathbf{T}^\top = \begin{bmatrix} e^{\lambda_1 t} & 0 & \cdots & 0 \\ 0 & e^{\lambda_2 t} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & e^{\lambda_n t} \end{bmatrix}$$

- When $\mathrm{Re}(\lambda_i) < 0$ we have

$$\lim_{t\to\infty} \mathbf{x}(t) = 0$$

- **Asymptotic stability:** If the real parts of all eigenvalues are negative, then the (LTI*) system is asymptotically stable.

*Linear Time Invariant

Controller

- State equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

- **State feedback**

$$\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t)$$

- After feedback, closed loop system

$$\dot{\mathbf{x}}(t) = (\mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{x}(t)$$

- Stability can be obtained by selecting an appropriate $\mathbf{K}$.

Observer

- State equation

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) \end{aligned}$$

- Copy of system

$$\dot{\mathbf{z}}(t) = \mathbf{A}\mathbf{z}(t) + \mathbf{B}\mathbf{u}(t)$$

- Difference between actual and copy systems

$$\begin{aligned} \dot{\mathbf{x}}(t) - \dot{\mathbf{z}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) - \mathbf{A}\mathbf{z}(t) - \mathbf{B}\mathbf{u}(t) \\ &= \mathbf{A}(\mathbf{x}(t) - \mathbf{z}(t)) \end{aligned}$$

- Let $\boldsymbol{\xi}(t) = \mathbf{x}(t) - \mathbf{z}(t)$. If $\mathbf{A}$ is stable,

$$\lim_{t\to\infty} \|\boldsymbol{\xi}(t)\| = \lim_{t\to\infty} \|e^{\mathbf{A}t}\boldsymbol{\xi}(0)\| = \mathbf{0}$$

Observer

- Even if $\mathbf{A}$ is not stable, the state can be estimated by using error feedback.
- State equation

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t)\end{aligned}$$

- Estimation error feedback

$$\dot{\mathbf{z}}(t) = \mathbf{A}\mathbf{z}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}(\mathbf{y}(t) - \mathbf{C}\mathbf{z}(t))$$

- Feedback system

$$\begin{aligned}\dot{\mathbf{x}}(t) - \dot{\mathbf{z}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) - \mathbf{A}\mathbf{z}(t) - \mathbf{B}\mathbf{u}(t) - \mathbf{G}(\mathbf{y}(t) - \mathbf{C}\mathbf{z}(t)) \\ &= (\mathbf{A} - \mathbf{G}\mathbf{C})(\mathbf{x}(t) - \mathbf{z}(t))\end{aligned}$$

- Stability can be obtained by appropriately selecting $\mathbf{G}$.

Nonlinear system

- Autonomous system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{0}) = \mathbf{f}(\mathbf{x}(t))$$

- State feedback

$$\mathbf{u}(t) = \phi(\mathbf{x}(t))$$

- After feedback

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \phi(\mathbf{x}(t)))$$

- This is also an autonomous system

Stability of nonlinear system

- **Equilibrium point**: $\mathbf{x}_e$ where $\mathbf{f}(\mathbf{x}_e(t)) = \mathbf{0}$.
  1. Local stability: Starting from a state sufficiently close to $\mathbf{x}_e$ then the solution will stay close to $\mathbf{x}_e$.
  2. Local asymptotic stability: Starting from a state sufficiently close to $\mathbf{x}_e$ then the state will asymptotically converge to $\mathbf{x}_e$.
  3. Exponential stability: Starting from a state sufficiently close to $\mathbf{x}_e$ then the state will converge to $\mathbf{x}_e$ exponentially.
  4. Global stability: Starting from any state then the state will stay close to $\mathbf{x}_e$.
  5. Global asymptotic stability: Starting from any state then the state will asymptotically converge to $\mathbf{x}_e$.

Stability of linearized system

- **Equilibrium point**: $\mathbf{x}_e$ where $\mathbf{f}(\mathbf{x}_e) = \mathbf{0}$
- Taylor expansion

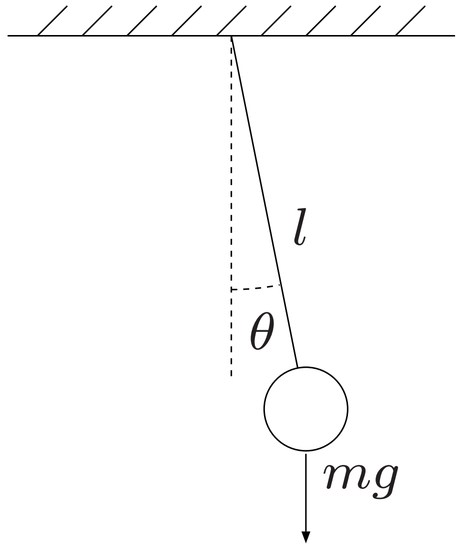
$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}_e) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_e} (\mathbf{x} - \mathbf{x}_e) + O((\mathbf{x} - \mathbf{x}_e)^2)$$

- Neglecting higher order terms

$$\dot{\bar{\mathbf{x}}} = \mathbf{A}\bar{\mathbf{x}}, \quad \mathbf{A} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_e}$$

- If this linearized system is stable then the original nonlinear system is locally asymptotically stable.

Example of pendulum



- mass: m, friction coefficient: γ, pendulum length: l
- Dynamical equation

$$ml\ddot{\theta} + \gamma\dot{\theta} + mg\sin\theta = 0$$

State equation of pendulum

- Dynamical equation:

$$ml\ddot{\theta} + \gamma\dot{\theta} + mg\sin\theta = 0$$

- State: $x_1 = \theta,\ x_2 = \dot{\theta}$

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{\gamma}{ml}x_2 - \frac{g}{l}\sin x_1\end{aligned}$$

- Equilibrium points: $\dot{\mathbf{x}} = [\dot{x}_1\ \ \dot{x}_2]^\top = [0\ \ 0]^\top$, i.e.,

$$x_2 = 0,\ \sin x_1 = 0$$

- Linearize the system at $\mathbf{x} = [0\ \ 0]^\top$ and $\mathbf{x} = [\pi\ \ 0]^\top$.

Linearized system at $x_1 = 0$

- **When** $x_1 = 0$

  $\cos x_1 = 1$ and we have

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\bigg|_{\mathbf{x}_0} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}\Bigg|_{(0,0)}$$

$$= \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & -\frac{\gamma}{ml} \end{bmatrix}$$

$$\det(s\mathbf{I} - \mathbf{A}) = s^2 + bs + c = s^2 + \frac{\gamma}{ml}s + \frac{g}{l}$$

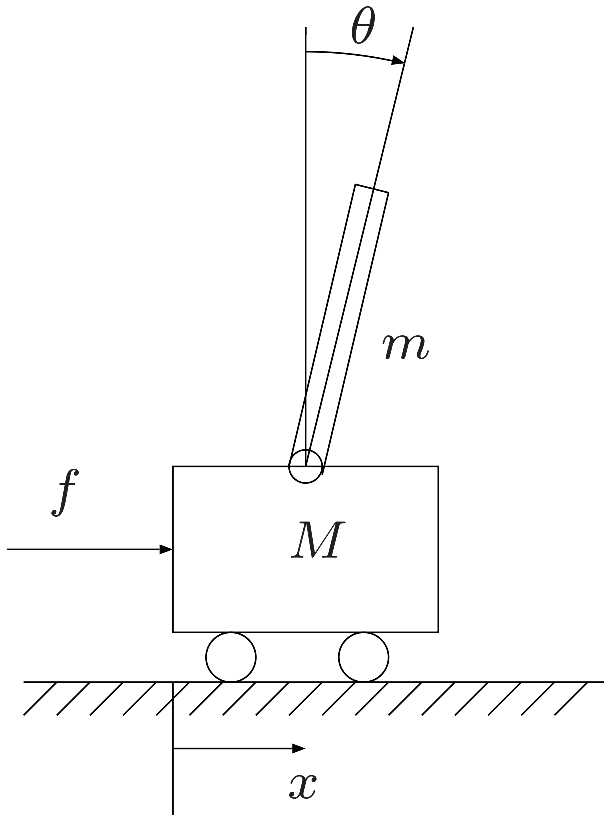
  Since $b > 0$, $c > 0$, $\mathbf{A}$ is stable.

Linearized system at $x_1 = \pi$

- **When** $x_1 = \pi$
  $\cos x_1 = -1$ and we have
  $$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ \frac{g}{l} & -\frac{\gamma}{ml} \end{bmatrix}$$
  $$\det(s\mathbf{I} - \mathbf{A}) = s^2 + bs + c = s^2 + \frac{\gamma}{ml}s - \frac{g}{l}$$
  Since $c < 0$, $\mathbf{A}$ is unstable.

- This example shows that the pendulum down position is stable and pendulum up position is unstable.
- When $\gamma = 0$, the eigenvalue of $\mathbf{A}$ becomes pure imaginary and the system is marginally stable (not asymptotically stable).

Control of nonlinear system



Modeling

- cart mass: M, pendulum mass: m, pendulum length: $2l$
- Internal force: horizontal: F_x, vertical: F_y
- cart position: x, pendulum angle: θ

$$I\ddot{\theta} + \mu_\theta \dot{\theta} = F_x l \cos\theta + F_y l \sin\theta, \qquad I = \frac{1}{3} m l^2$$

- F_x and $F_y - mg$ generates pendulum acceleration.

$$\begin{aligned} F_x &= m\frac{d^2}{dt^2}(x - l\sin\theta) \\ F_y - mg &= m\frac{d^2}{dt^2}(l\cos\theta) \end{aligned}$$

- $f - F_x$ generates cart acceleration.

$$f - F_x = M\ddot{x} + \mu_x \dot{x}$$

Modeling

- Eliminate the internal force, we have

$$
\begin{aligned}
(M+m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta + \mu_x\dot{x} &= f \\
(I+ml^2)\ddot{\theta} + ml\ddot{x}\cos\theta - mgl\sin\theta + \mu_\theta\dot{\theta} &= 0
\end{aligned}
$$

Linearization

- Assume that θ is small.
- Approximation

$$\sin\theta \approx \theta, \cos\theta \approx 1$$

- Linearized system

$$\begin{aligned}(M+m)\ddot{x} + ml\ddot{\theta} + \mu_x\dot{x} &= f \\ (I+ml^2)\ddot{\theta} + ml\ddot{x} - mgl\theta + \mu_\theta\dot{\theta} &= 0\end{aligned}$$

- Neglect force from pendulum to cart

$$\begin{aligned}\ddot{x} + \hat{\mu}_x\dot{x} &= \alpha f, \qquad \alpha = \frac{1}{M+m}, \\ \ddot{\theta} + \hat{\mu}_\theta\dot{\theta} &= -\beta\ddot{x} + \beta g\theta, \qquad \beta = \frac{ml}{I+ml^2}, \\ \hat{\mu}_x &= \frac{\mu_x}{M+m}, \qquad \hat{\mu}_\theta = \frac{\mu_\theta}{I+ml^2}\end{aligned}$$

State equation

- State equation

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u$$

where $\mathbf{x} = [x \ \theta \ \dot{x} \ \dot{\theta}]^\top$ and

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\widehat{\mu}_x & 0 \\ 0 & \beta g & \beta\widehat{\mu}_x & -\widehat{\mu}_\theta \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \alpha \\ -\alpha\beta \end{bmatrix}$$

Controller design

- Assume that the state $\mathbf{x} = [x\ \theta\ \dot{x}\ \dot{\theta}]^\top$ is available, then state feedback

$$u = -\mathbf{K}\mathbf{x}$$

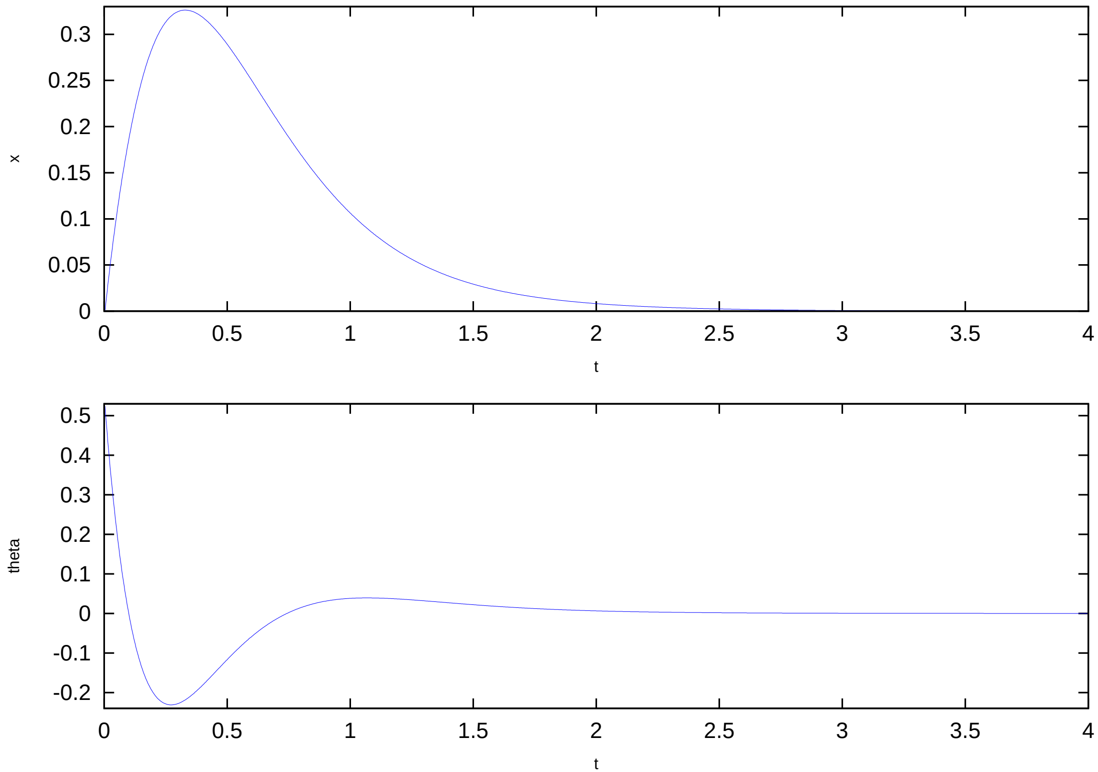
  is used.
- LQ regulator that minimize

$$J = \int_0^\infty \left\{ \mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u} \right\} dt$$

  can be designed using `lqr` command of Matlab or octave.

Simulation

- $\alpha = 90, \ \beta = 3.7, \ \hat{\mu}_x = 240, \ \hat{\mu}_\theta = 0.02$
- $\mathbf{R} = 1, \mathbf{Q} = \text{diag}[50, 1, 1, 1]$
- $\mathbf{K} = [-7.07, -15.75, -8.21, -2.80]$



Observer

- When only cart position and pendulum angle are available,

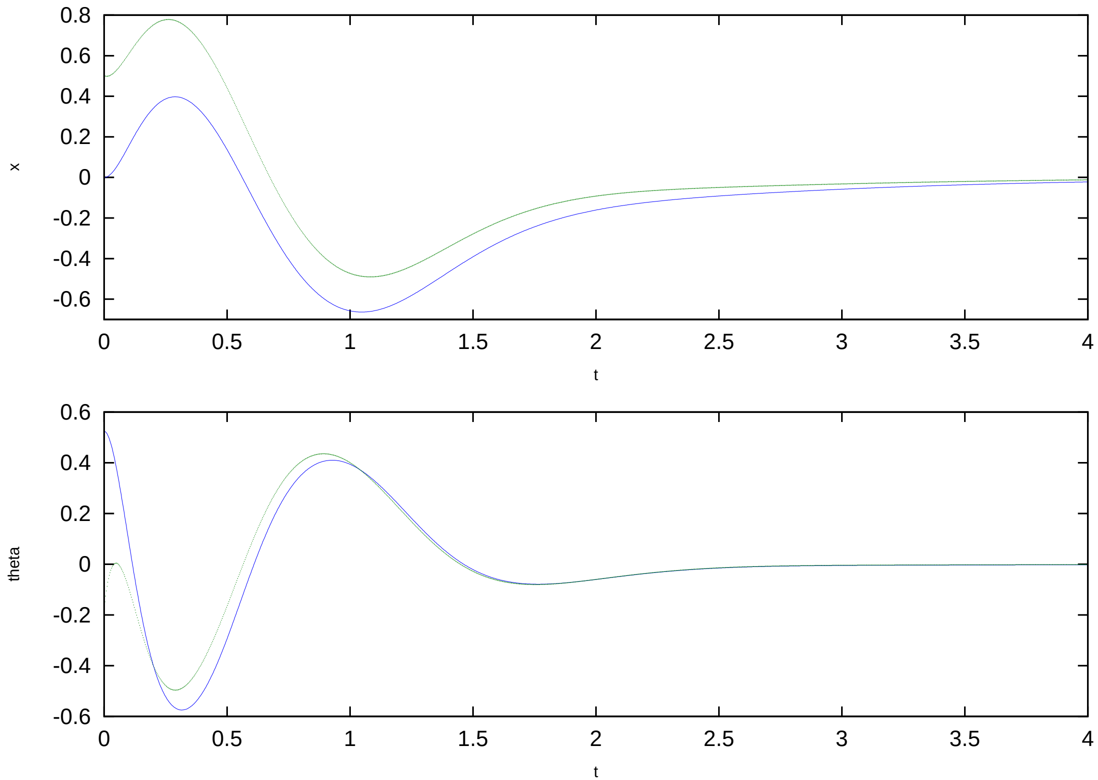
$$\mathbf{y} = \begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x}$$

- `lqe` command of Matlab or octave is used to obtain

$$\mathbf{G} = \begin{bmatrix} 1.00 & 3.11e-3 \\ 3.11e-4 & 1.21 \\ 8.64e-6 & 3.06e-5 \\ 4.04e-3 & 72.5 \end{bmatrix}$$

Simulation

- $\mathbf{x}(0) = [0\ 0.52\ 0\ 0]^\top$
- $\mathbf{z}(0) = [0.5\ -0.174\ 0\ 0]^\top$
- Observer feedback: $u = -\mathbf{K}\mathbf{z}$



Lyapunov Function

- **Lyapunov's method** is to check stability of nonlinear system by investigating the (generalized) energy of the system.
- Nonlinear autonomous system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t))$$

- Suppose that equilibrium point $\mathbf{x}_e$ is $\mathbf{0}$.
- If $\mathbf{x}_e$ is not $\mathbf{0}$ then put $\mathbf{z}(t) = \mathbf{x}(t) - \mathbf{x}_e$ and check the stability of

$$\dot{\mathbf{z}} = \mathbf{f}(\mathbf{x}(t)) = \mathbf{f}(\mathbf{z}(t) + \mathbf{x}_e)$$

Positive definite function

- State $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$
- Scalar valued function

$$V(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$$

is said positive definite if it satisfies the following in region Ω which includes $\mathbf{0}$

1. $V(\mathbf{0}) = 0$
2. For any $\mathbf{x} \in \Omega$ $(\mathbf{x} \neq \mathbf{0})$, $V(\mathbf{x}) > 0$

- $V(\mathbf{x}) \geq 0$: positive semi-definite
- $V(\mathbf{x}) < 0$: negative definite
- $V(\mathbf{x}) \leq 0$: negative semi-definite

Derivative along trajectory

- Derivative of $V(\mathbf{x})$ along its solution trajectory is defined as follows

$$\dot{V}(\mathbf{x}) = \frac{dV(\mathbf{x}(t))}{dt} = \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}}\dot{\mathbf{x}} = \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})$$

where

$$\frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial V}{\partial x_1} & \frac{\partial V}{\partial x_2} & \cdots & \frac{\partial V}{\partial x_n} \end{bmatrix}$$

- Thus it is actually the inner product of gradient of $V(\mathbf{x})$ and $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$.

$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial x_1}f_1(\mathbf{x}) + \frac{\partial V}{\partial x_2}f_2(\mathbf{x}) + \cdots + \frac{\partial V}{\partial x_n}f_n(\mathbf{x})$$

Lyapunov stability theorem

- Sufficient condition of local stability

  (LS1) $V(\mathbf{x})$ is positive definite in Ω

  (LS2) $\dot{V}(\mathbf{x})$ is negative semi-definite in Ω

- Sufficient condition of local asymptotic stability

  (LAS1) $V(\mathbf{x})$ is positive definite in Ω

  (LAS2) $\dot{V}(\mathbf{x})$ is negative definite in Ω

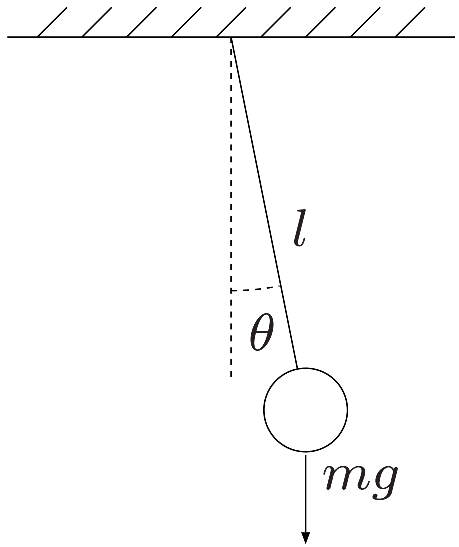
- Sufficient condition of global asymptotic stability

  (GAS1) $V(\mathbf{0}) = 0$ and $V(\mathbf{x}) > 0 \quad \forall \mathbf{x} \neq 0$

  (GAS2) $\dot{V}(\mathbf{x}) < 0 \quad \forall \mathbf{x} \neq 0$

  (GAS3) When $\|\mathbf{x}\| \to \infty$, then $V(\mathbf{x}) \to \infty$

Pendulum example



- $x_1 = \theta, \; x_2 = \dot{\theta}$

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{\gamma}{ml}x_2 - \frac{g}{l}\sin x_1 \end{bmatrix} = \mathbf{f}(\mathbf{x})$$

Pendulum example

- Lyapunov function candidate: total energy

$$\begin{aligned} V &= K + P = \frac{1}{2}m(\omega l)^2 + mgh \\ &= \frac{1}{2}ml^2x_2^2 + mgl(1-\cos x_1) \end{aligned}$$

- In this case $V(\mathbf{0}) = 0$ is satisfied by $x_2 = 0$, $\cos x_1 = 1$.
- $\Omega = [(-\pi, \pi), \mathbb{R}]$
- Then $V(\mathbf{x})$ is positive definite in Ω.
- Derivative of V

$$\begin{aligned} \dot{V}(x) &= \begin{bmatrix} \dfrac{\partial V}{\partial x_1} & \dfrac{\partial V}{\partial x_2} \end{bmatrix} \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} \\ &= \begin{bmatrix} mg\ell \sin x_1 & m\ell^2 x_2 \end{bmatrix} \begin{bmatrix} x_2 \\ -\frac{\gamma}{l}x_2 - \frac{g}{\ell}\sin x_1 \end{bmatrix} \\ &= \gamma l x_2^2 \leq 0 \end{aligned}$$

Pendulum example

- Thus the equilibrium point $\mathbf{x} = [0\ 0]^\top$ is locally stable.

- To show the asymptotic stability we need **LaSalle theorem**.
- The invariant set of state that satisfy

$$\dot{V} = \gamma l x_2^2 = 0$$

is $x_2 = 0, \dot{x}_2 = 0$.
- Since

$$\dot{x}_2 = -\frac{\gamma}{ml} x_2 - \frac{g}{l} \sin x_1,$$

$\sin x_1 = 0$ is concluded, and in Ω this is satisfied only by $x_1 = 0$.

Robot kinematics

- **Kinematics:** The end tip position $\mathbf{r}$ of the robot is a function of joint angle $\boldsymbol{\theta}$.

$$\mathbf{r} = \mathbf{f}(\boldsymbol{\theta})$$

- **Inverse Kinematics:** To find a set of joint angle $\boldsymbol{\theta}^*$ that satisfy a specified end tip position $\mathbf{r}^*$. Formally it is written as

$$\boldsymbol{\theta}^* = \mathbf{f}^{-1}(\mathbf{r}^*),$$

but difficult to find a general solution.

Jacobi matrix

- Taylor expansion of $\mathbf{f}$

$$\mathbf{r}^* - \mathbf{r} = \mathbf{f}(\boldsymbol{\theta}^*) - \mathbf{f}(\boldsymbol{\theta}) = \mathbf{J}\Delta\boldsymbol{\theta} + O((\Delta\boldsymbol{\theta})^2)$$

- Iterative solution of inverse kinematics:

$$\Delta\boldsymbol{\theta} = \mathbf{J}^{-1}(\mathbf{r}^* - \mathbf{r})$$

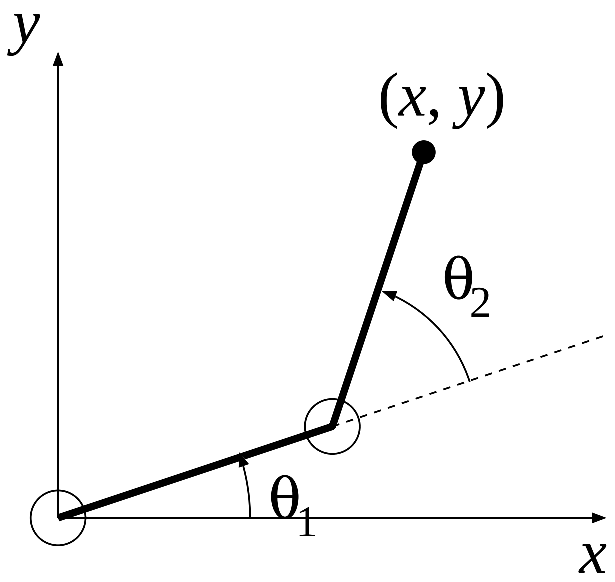
  where $\mathbf{J}$ is defined by

$$\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}}$$

  and called **Jacobi matrix**.

- Jacobi matrix is a function of joint angle $\boldsymbol{\theta}$.

Robot Kinematics (2 link example)



- Endtip Position:

$$\begin{aligned} x &= \ell_1 \cos\theta_1 + \ell_2 \cos(\theta_1+\theta_2) \\ y &= \ell_1 \sin\theta_1 + \ell_2 \sin(\theta_1+\theta_2) \end{aligned}$$

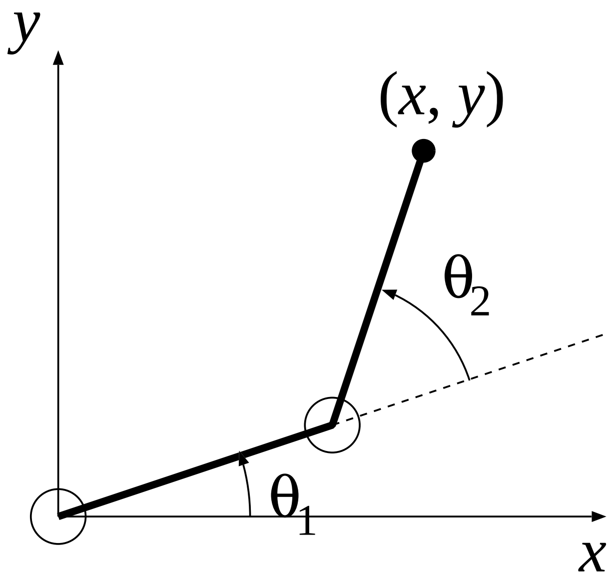
- Suppose that the motor driver is velocity control, i.e.,

$$u_1 = \dot\theta_1, \quad u_2 = \dot\theta_2$$

- Dynamical Equation:

$$\begin{aligned} \dot x &= -\ell_1 \sin\theta_1 \dot\theta_1 \\ &\quad -\ell_2 \sin(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \\ \dot y &= \ell_1 \cos\theta_1 \dot\theta_1 \\ &\quad +\ell_2 \cos(\theta_1+\theta_2)(\dot\theta_1+\dot\theta_2) \end{aligned}$$

Robot Kinematics (2 link example)



- Dynamical Equation:

$$\begin{aligned}\dot{x} &= -\ell_1 \sin\theta_1 \dot{\theta}_1 \\ &\quad -\ell_2 \sin(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2) \\ \dot{y} &= \ell_1 \cos\theta_1 \dot{\theta}_1 \\ &\quad +\ell_2 \cos(\theta_1+\theta_2)(\dot{\theta}_1+\dot{\theta}_2)\end{aligned}$$

- State Equation:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -\ell_1 \sin\theta_1\dot{\theta}_1 - \ell_2\sin(\theta_1+\theta_2) & -\ell_2\sin(\theta_1+\theta_2) \\ \ell_1\cos\theta_1\dot{\theta}_1 + \ell_2\cos(\theta_1+\theta_2) & \ell_2\cos(\theta_1+\theta_2) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

- No $f(x)$.
- The robot system has **kinematic nonlinearity**.

Robot Kinematics (2 link example)

- Let $r = [x, y]^\top$ be the output, then the system is described by

$$\mathbf{r} = \mathbf{f}(\boldsymbol{\theta})$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2]^\top$.

- Then we have

$$\dot{\mathbf{r}} = \mathbf{J}(\boldsymbol{\theta})\mathbf{u},$$

where

$$\mathbf{u} = \dot{\boldsymbol{\theta}} \quad \text{and} \quad \mathbf{J} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \frac{\partial f_1}{\partial \theta_2} \\ \frac{\partial f_2}{\partial \theta_1} & \frac{\partial f_2}{\partial \theta_2} \end{bmatrix}.$$

The matrix $\mathbf{J}$ is called **Jacobi matrix**.

Resolved motion rate control

- Objective: $\mathbf{r}(t) \rightarrow \mathbf{r}^*(t)$
- Derivative relation:

$$\dot{\mathbf{r}}(t) = \mathbf{J}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t)$$

- A simple control law:

$$\dot{\boldsymbol{\theta}}(t) = \lambda \mathbf{J}^{-1}(\boldsymbol{\theta}(t))(\mathbf{r}^* - \mathbf{r}(t))$$

  or

$$\dot{\boldsymbol{\theta}}(t) = \lambda \mathbf{J}^{-1}(\boldsymbol{\theta}(t))(\mathbf{r}^* - \mathbf{f}(\boldsymbol{\theta}(t)))$$

- Resolved Motion Rate Control (Whitney, 1969)

Stability check (RMRC)

- Stability at $\boldsymbol{\theta} = \boldsymbol{\theta}^*$
- System

$$\dot{\mathbf{r}}(t) = \mathbf{J}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}}(t)$$

- Control law

$$\dot{\boldsymbol{\theta}}(t) = \lambda \mathbf{J}^{-1}(\boldsymbol{\theta}(t))(\mathbf{r}^* - \mathbf{r}(t))$$

- Closed loop dynamics

$$\dot{\mathbf{r}}(t) = \lambda \mathbf{J}(\boldsymbol{\theta}(t))\mathbf{J}^{-1}(\boldsymbol{\theta}(t))(\mathbf{r}^* - \mathbf{r}(t)) = \lambda(\mathbf{r}^* - \mathbf{r}(t))$$

- Let $\mathrm{e}(t) = \mathbf{r}(t) - \mathbf{r}^*$ then we have

$$\dot{\mathrm{e}}(t) = -\lambda \mathbf{e}(t)$$

$$\mathbf{e}(t) = \mathrm{e}(0)\exp(-\lambda t)$$

Fixed gain control law

- In RMRC, $\mathbf{J}(\boldsymbol{\theta}(t))$ and its inverse have to be computed in realtime.
- Instead, fixed gain matrix $\mathbf{J}^* = \mathbf{J}(\boldsymbol{\theta}^*)$ can also be used.

$$\dot{\boldsymbol{\theta}}(t) = \lambda \mathbf{J}^{*-1}(\mathbf{r}^* - \mathbf{r}(t))$$

Stability check (Fixed gain)

- Closed loop system

$$\dot{\mathbf{r}}(t) = \lambda \mathbf{J}(\boldsymbol{\theta}(t))\mathbf{J}^{*-1}(\mathbf{r}^* - \mathbf{r}(t))$$

- **Equilibrium point**: When $\boldsymbol{\theta} = \boldsymbol{\theta}^*$, $\mathbf{r} = \mathbf{r}^*$ and $\dot{\mathbf{r}} = \mathbf{0}$. Thus $\boldsymbol{\theta} = \boldsymbol{\theta}^*$ is an equilibrium point.

- **Lyapnov function candidate**:

$$V(t) = (\mathbf{r}^* - \mathbf{r}(t))^\top (\mathbf{r}^* - \mathbf{r}(t))$$

- Derivative along the trajectory

$$\dot{V}(t) = -2(\mathbf{r}^* - \mathbf{r}(t))^\top \dot{\mathbf{r}}(t)$$

Stability check (Fixed gain)

- Thus we have

$$\dot{V}(t) = -2\lambda(\mathbf{r}^* - \mathbf{r}(t))^\top \mathbf{J}(\boldsymbol{\theta}(t))\mathbf{J}^{*-1}(\mathbf{r}^* - \mathbf{r}(t))$$

- At the equilibrium point $\boldsymbol{\theta} = \boldsymbol{\theta}^*$, since $\mathbf{r} = \mathbf{r}^*$. Thus we have $\dot{V} = 0$.
- Also in the neighborhood of the equilibrium point,

$$\mathbf{J}(\boldsymbol{\theta})\mathbf{J}^{*-1} \approx \mathbf{J}^*\mathbf{J}^{*-1} = \mathbf{I}$$

  and thus we have $\dot{V} < 0$.
- The region in which $\dot{V} < 0$ holds is not explicitly given.

Efficient second-order minimization

- Let $\mathbf{J} = \mathbf{J}(\boldsymbol{\theta}(t))$ and compute

$$\mathbf{J}_{\mathsf{esm}} = (\mathbf{J} + \mathbf{J}^*)/2$$

in realtime.

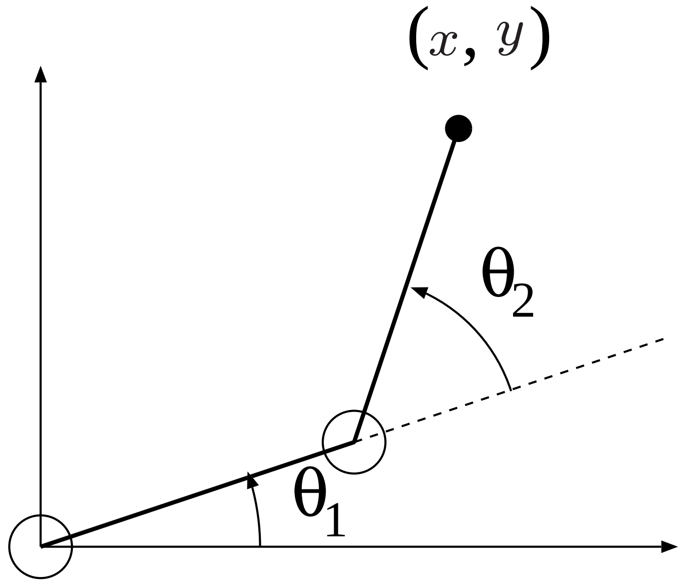
- Control law

$$\dot{\boldsymbol{\theta}}(t) = \lambda \mathbf{J}_{\mathsf{esm}}^{-1}(\mathbf{r}^* - \mathbf{r}(t))$$

- Since $\mathbf{J}_{\mathsf{esm}} = \mathbf{J}^*$ at the equilibrium point, the local stability property is the same as RMRC.
- This is efficient because $\mathbf{J}_{\mathsf{esm}}$ approximate the Taylor expansion of $\mathbf{r}$ to the second order.

$$\mathbf{r}^* - \mathbf{r} = \mathbf{J}_{\mathsf{esm}} \Delta\boldsymbol{\theta} + O((\Delta\boldsymbol{\theta})^3)$$

Two link robot



- Endtip position

$$\mathbf{r} = \begin{bmatrix} l_1 C_1 + l_2 C_{12} \\ l_1 S_1 + l_2 S_{12} \end{bmatrix}$$

where $C_1 = \cos\theta_1, S_1 = \sin\theta_1, C_{12} = \cos(\theta_1 + \theta_2), S_{12} = \sin(\theta_1 + \theta_2)$

Jacobi matrix

- Jacobi matrix

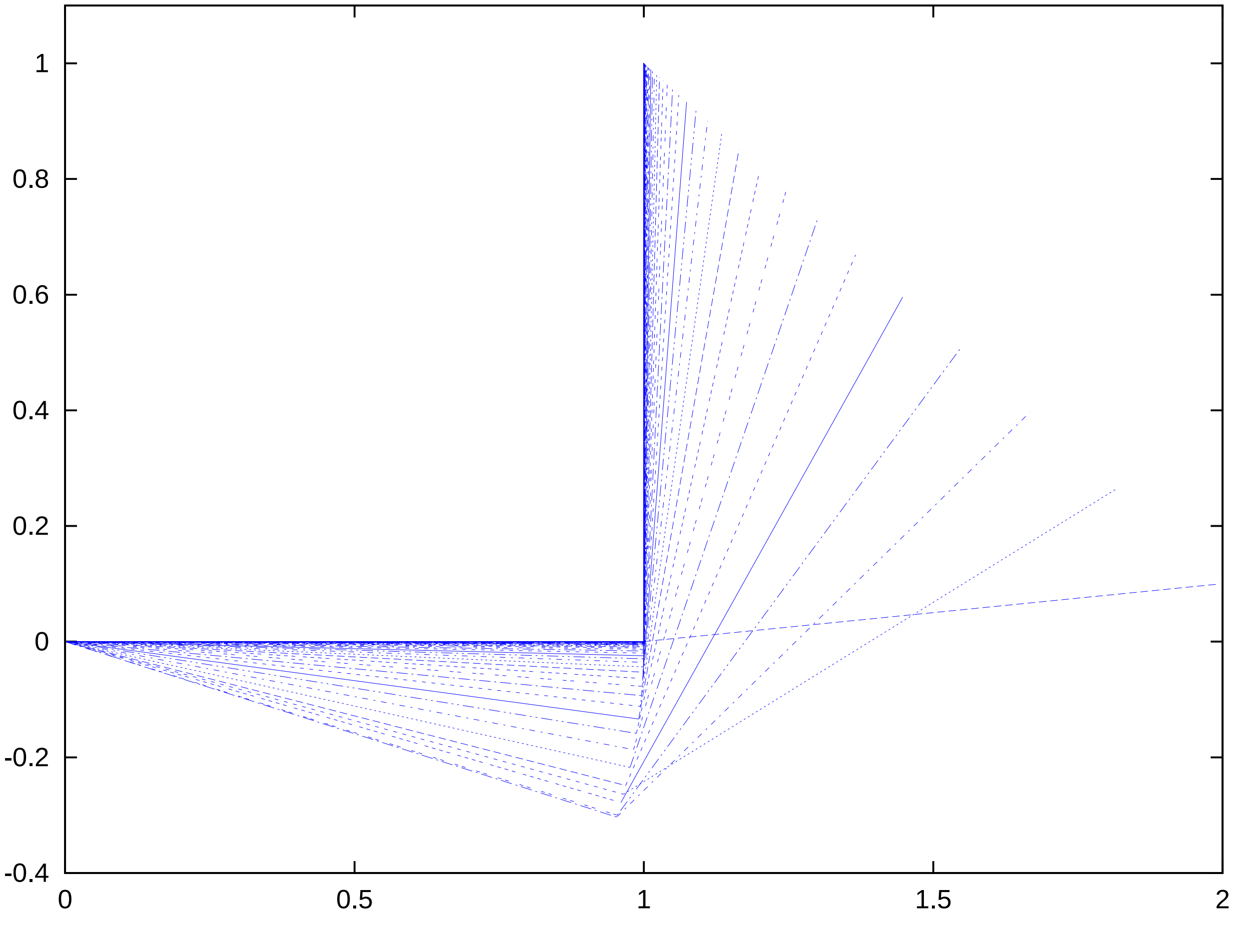
$$\mathbf{J} = \frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}} = \begin{bmatrix} -l_1 S_1 - l_2 S_{12} & -l_2 S_{12} \\ l_1 C_1 + l_2 C_{12} & l_2 C_{12} \end{bmatrix}$$

- Suppose that the link lengths are $l_1 = l_2 = 1$ and the desired tip position is $x = 1, y = 1$.
- There are two configurations that achieves this position. Here we assume that it is $\theta_1 = 0, \theta_2 = \pi/2$.
- Then the Jacobi matrix at the desired position is

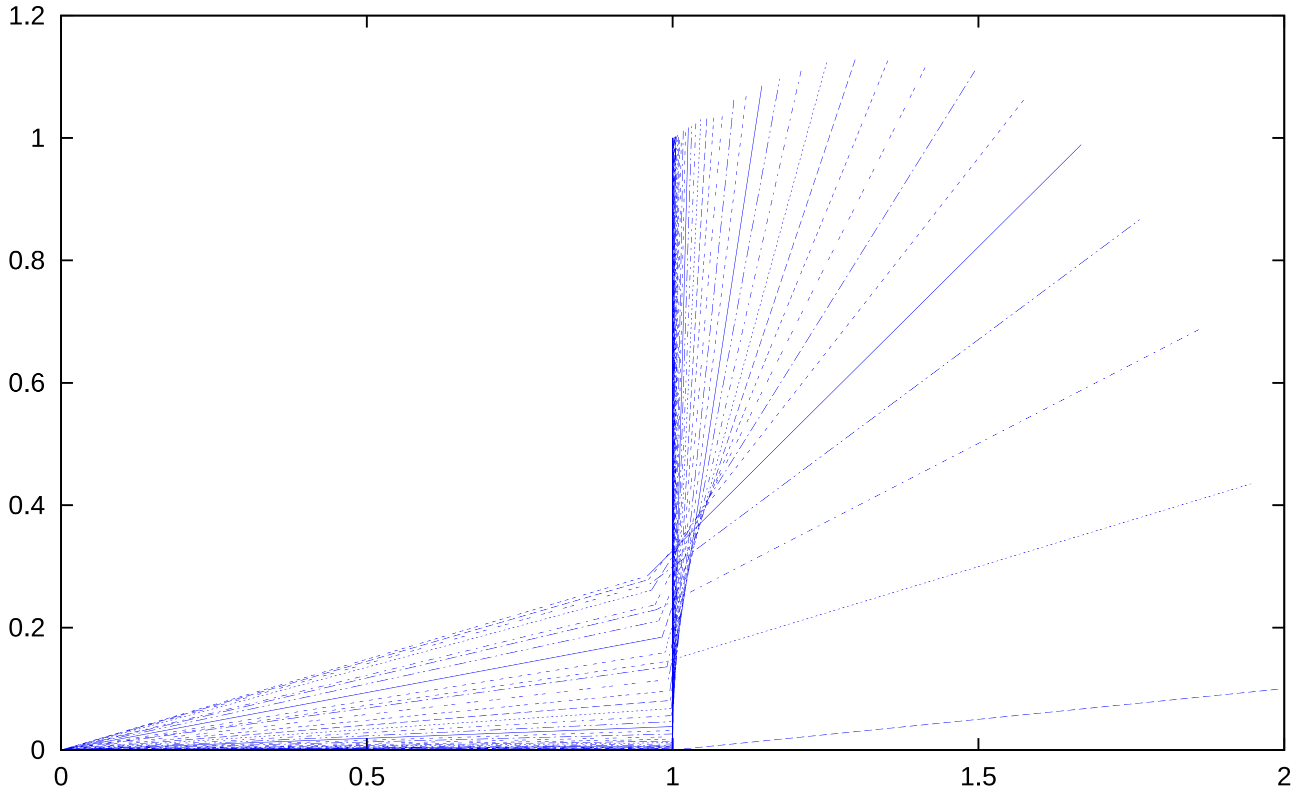
$$\mathbf{J}^* = \begin{bmatrix} -1 & -1 \\ 1 & 0 \end{bmatrix}$$

- Initial position is $\boldsymbol{\theta}_0 = [0\ 0.1]^\top$.

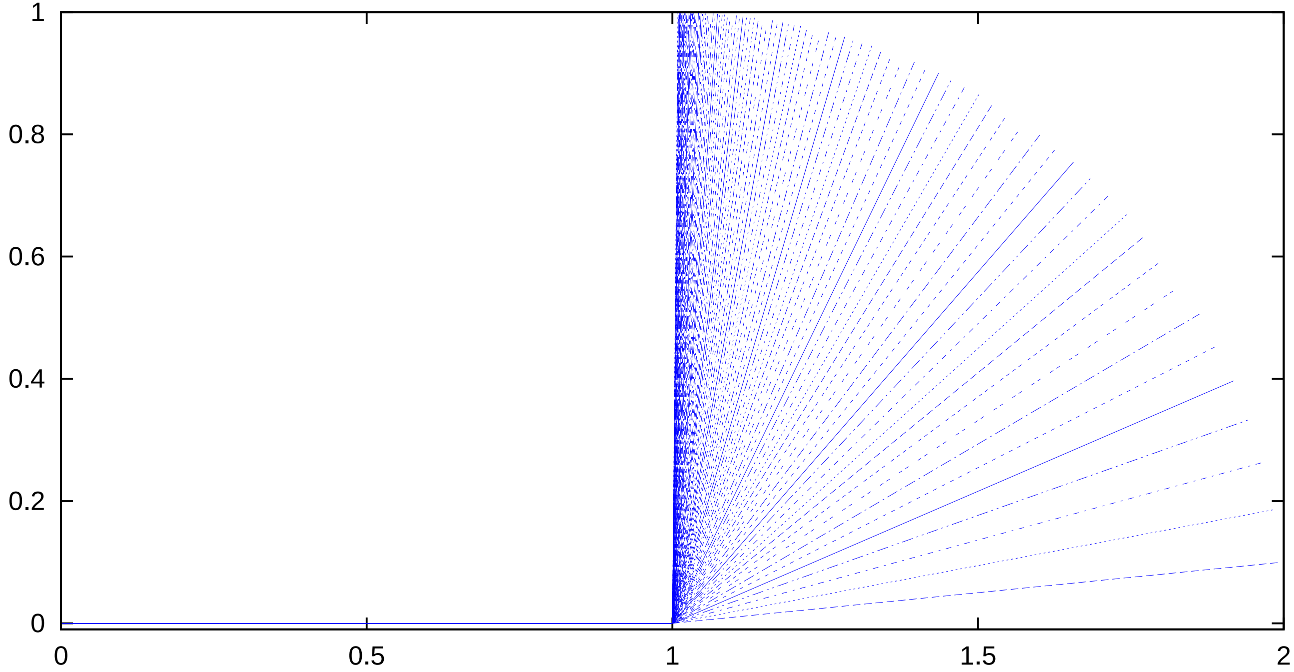
Simulations: RMRC



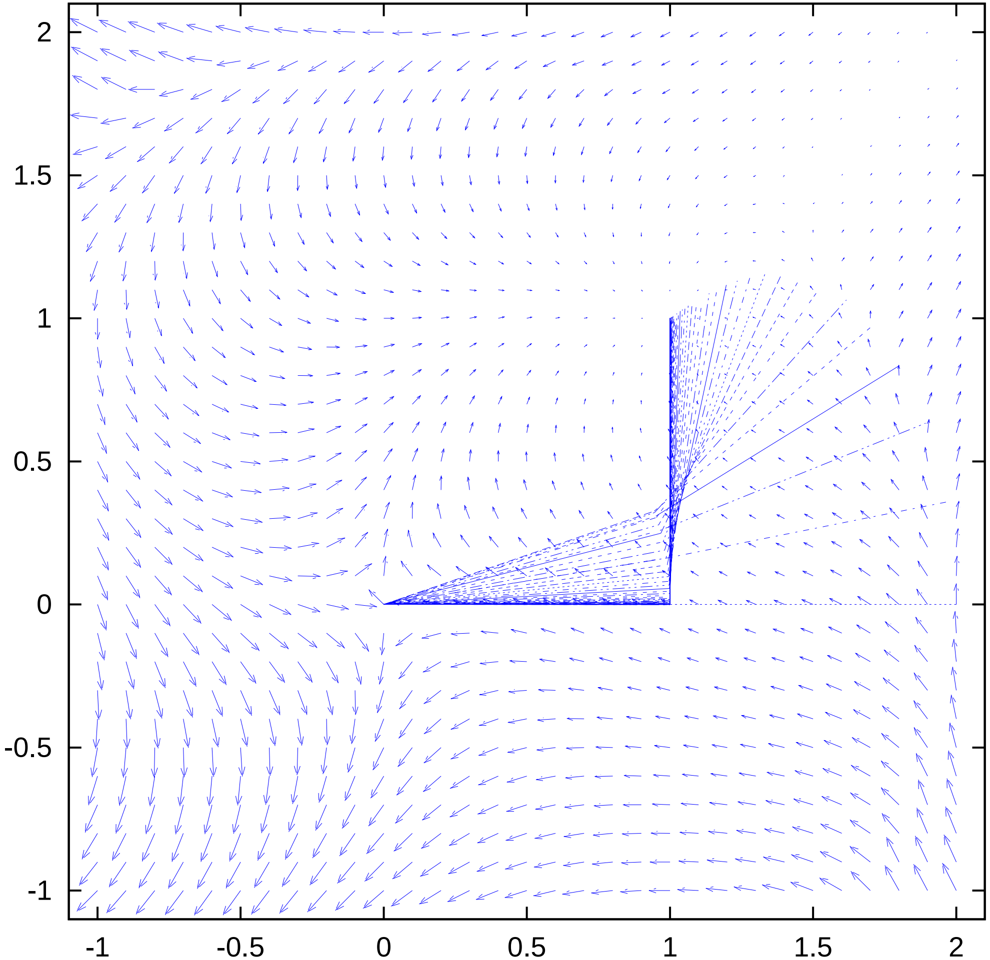
Simulations: Fixed gain



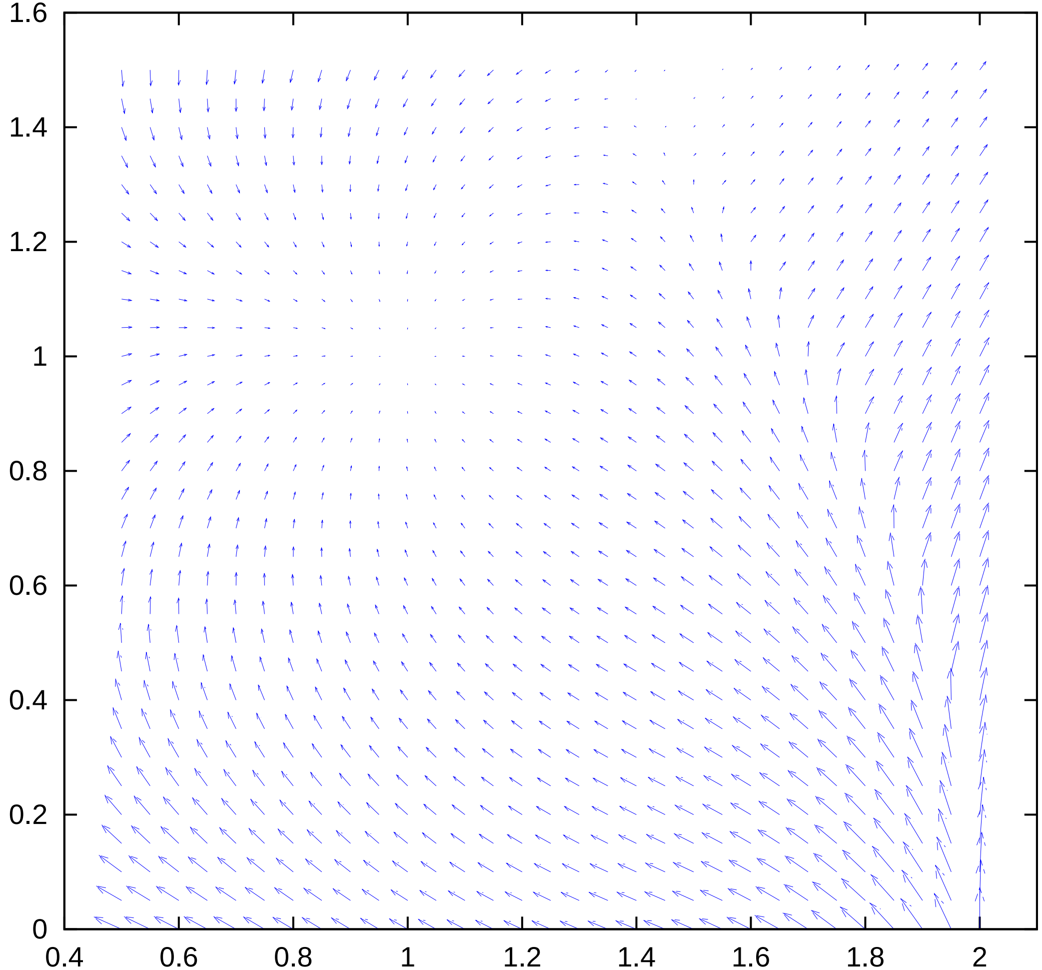
Simulations: ESM



Vector flow



Vector flow along the end tip trajectory



Resolved motion acceleration control

- Previous examples neglected the robot dynamics
- In general robot dynamics is given by

$$\mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\dot{\boldsymbol{\theta}}, \boldsymbol{\theta}) + \mathbf{G}(\boldsymbol{\theta}) = \boldsymbol{\tau}$$

  where $\mathbf{M}$ is inertia matrix, $\mathbf{C}$ is centrifugal and Coriolis force, and $\mathbf{G}$ is gravity force.
- Let the estimation of these parameters be $\hat{\mathbf{M}}$, $\hat{\mathbf{C}}$, $\hat{\mathbf{G}}$, respectively.
- Control law:

$$\boldsymbol{\tau} = \hat{\mathbf{M}}(\boldsymbol{\theta})\mathbf{v} + \hat{\mathbf{C}}(\dot{\boldsymbol{\theta}}, \boldsymbol{\theta}) + \hat{\mathbf{G}}(\boldsymbol{\theta})$$

  where

$$\mathbf{v} = \ddot{\boldsymbol{\theta}}^* + \lambda_1(\dot{\boldsymbol{\theta}}^* - \dot{\boldsymbol{\theta}}) + \lambda_2(\boldsymbol{\theta}^* - \boldsymbol{\theta})$$

  and λ_1, λ_2 are feedback gains.

Resolved motion acceleration control

- If the estimations are exact, we have following closed loop dynamics

$$\ddot{\mathbf{e}} + \lambda_1 \dot{\mathbf{e}} + \lambda_2 \mathbf{e} = 0, \quad \mathbf{e} = \boldsymbol{\theta}^* - \boldsymbol{\theta}$$

which is asymptotically stable.

- However it is not easy to obtain good estimations. If the parameter estimations are not correct, then the dynamics are not well canceled and the performance is deteriorated.
- This control law requires a lot of computations. Parallel algorithms and high speed approximations have been proposed.

Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- Sampling time issues
- ESM algorithm and visual tracking

Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- Sampling time issues
- ESM algorithm and visual tracking

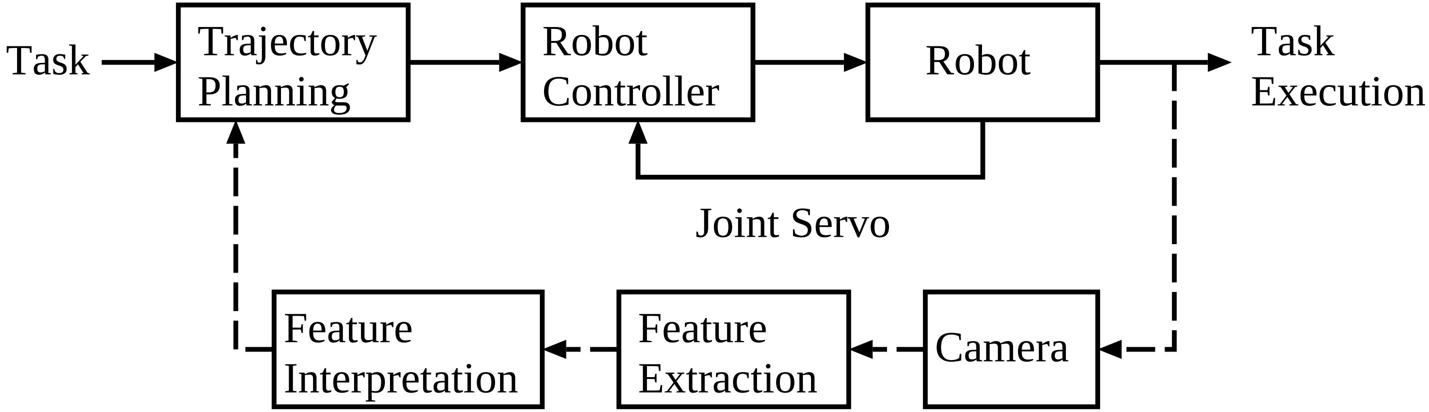
3D visual servo

- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

3D visual servo

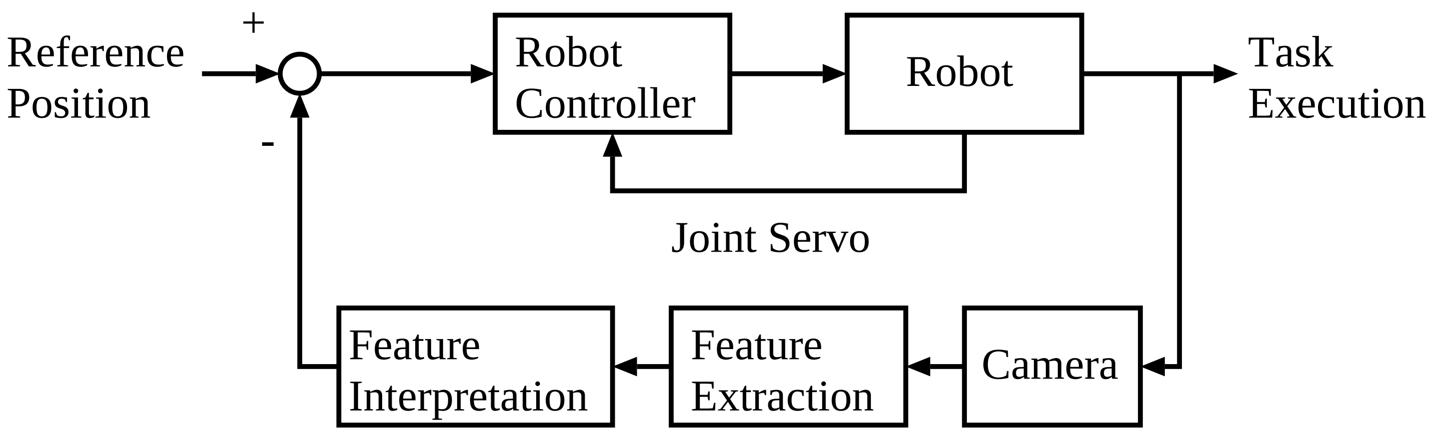
- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

Static visual servo, 1960-1975



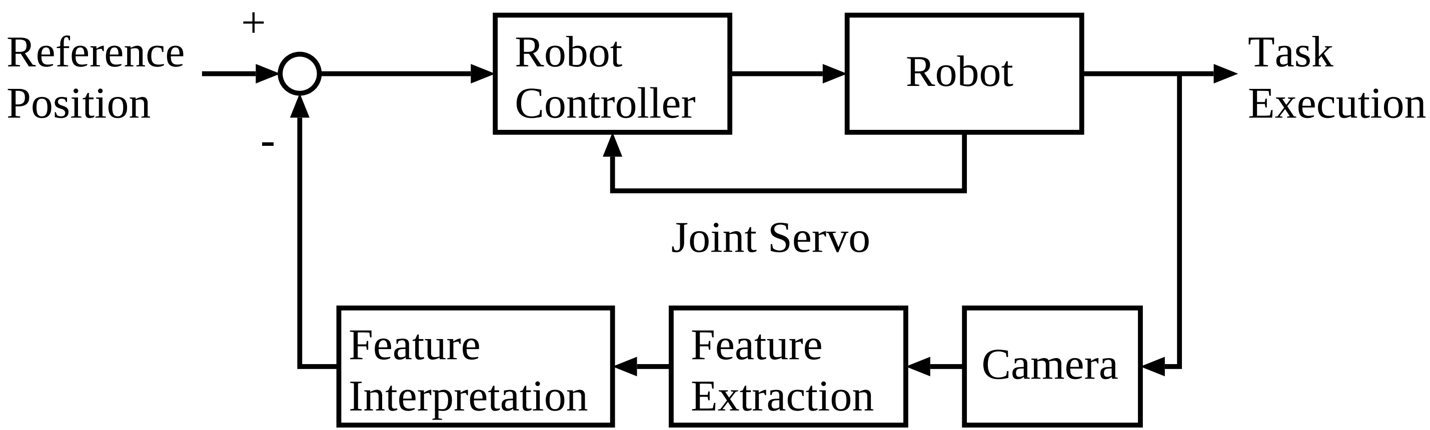
- Look and move
- Dashed line 0.1cycle/sec
- Only applicable to static object
- Position recognition for simple object

Position-based visual servo, 1975-1985



- Looking while moving
- Position recognition is done in realtime
- Special hardware for image processing is necessary
- Robust and fast position recognition is the key

Position-based visual servo, 1985-

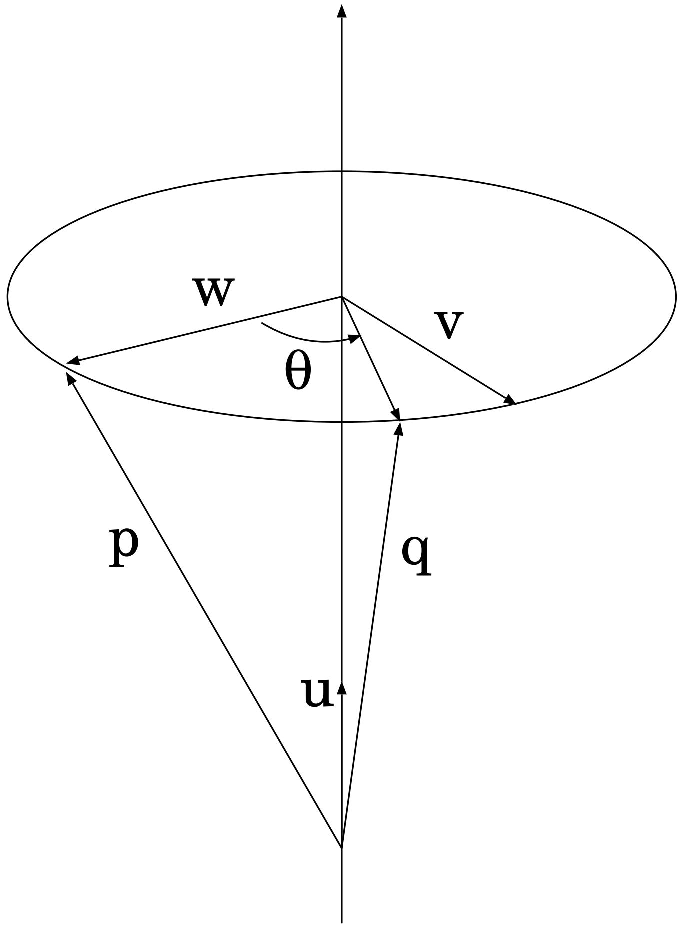


- Due to quick development of hardware, realtime image processing is available with CPU, GPU, multi-core...
- Realtime stereo is also possible
- In this section, position estimation is described

3D visual servo

- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

Expression of rotation

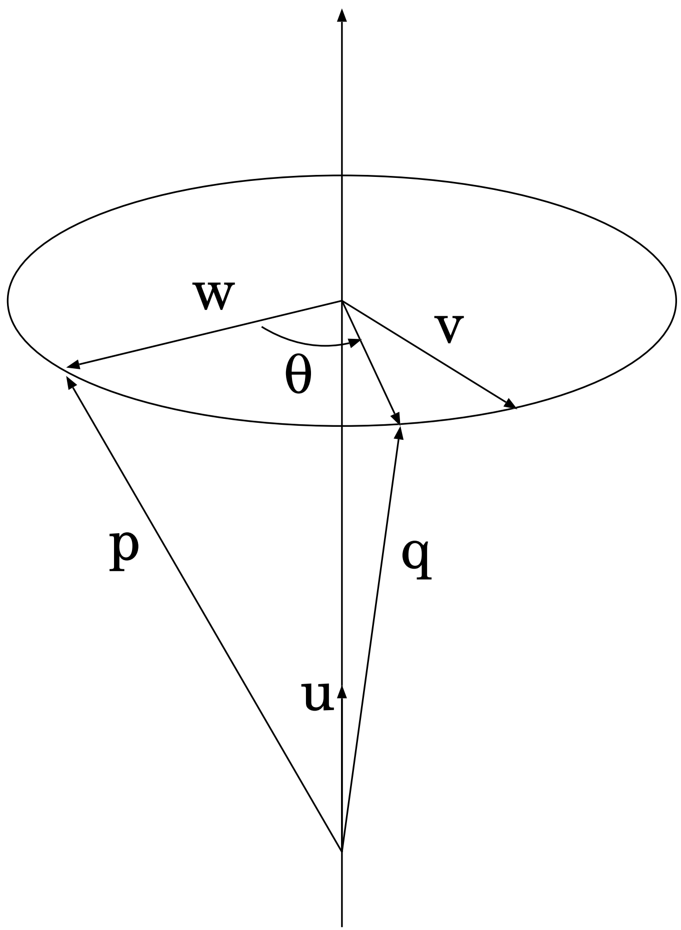


- $\mathbf{q}$ is a vector obtained by rotating $\mathbf{p}$ with rotation matrix $\mathbf{R}$.

$$\mathbf{q} = \mathbf{R}\mathbf{p}$$

- The rotation matrix is equivalent to a rotation of θ around the unit vector $\mathbf{u}$.
- Find the relationship between $\mathbf{R}$ and $(\theta, \mathbf{u})$.

Rodrigues formula



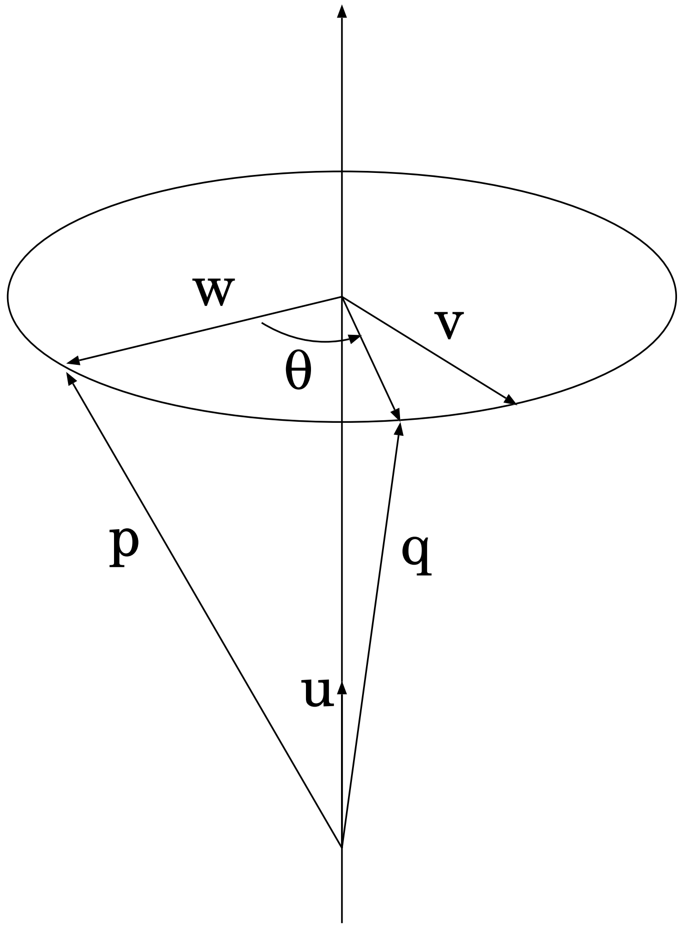
- Consider a circular disk orthogonal to $\mathbf{u}$ and contains points $\mathbf{p}$ and $\mathbf{q}$.
- Let $\mathbf{w}$ be the vector in the disk which is the projection of $\mathbf{p}$ onto the disk plane.
- Let $\mathbf{v}$ be the vector perpendicular to $\mathbf{w}$ in the disk.
- Then we have

$$\mathbf{q} = \alpha\mathbf{u} + \sin\theta\mathbf{v} + \cos\theta\mathbf{w}$$

where

$$\mathbf{w} = \mathbf{p} - \alpha\mathbf{u}, \quad \mathbf{v} = \mathbf{u} \wedge \mathbf{w}$$

Rodrigues formula



- Substituting $\mathbf{w}$ into right hand side of $\mathbf{q}$ yields

$$\begin{aligned}\mathbf{q} &= \alpha\mathbf{u} + \sin\theta\mathbf{v} + \cos\theta\mathbf{w} \\ &= \mathbf{p} - \mathbf{w} + \sin\theta(\mathbf{u}\wedge\mathbf{p}) + \cos\theta\mathbf{w} \\ &= \mathbf{p} + \sin\theta(\mathbf{u}\wedge\mathbf{p}) + (\cos\theta - 1)\mathbf{w}\end{aligned}$$

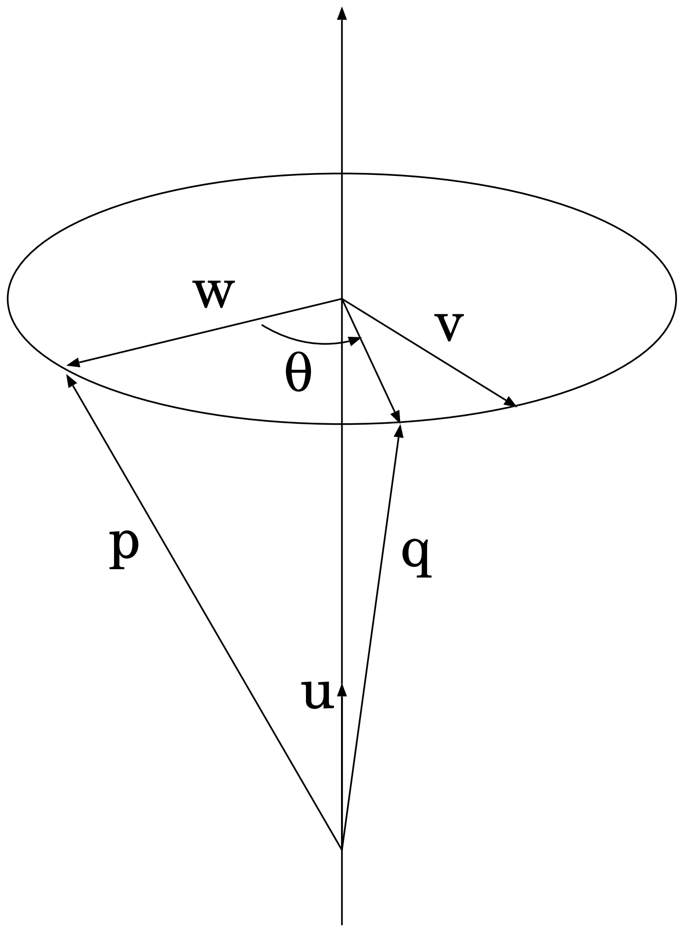
- Moreover

$$\mathbf{w} = -\mathbf{u}\wedge\mathbf{v} = -\mathbf{u}\wedge(\mathbf{u}\wedge\mathbf{p})$$

- Thus we have

$$\begin{aligned}\mathbf{q} &= \mathbf{p} + \sin\theta(\mathbf{u}\wedge\mathbf{p}) \\ &\quad + (1-\cos\theta)\mathbf{u}\wedge(\mathbf{u}\wedge\mathbf{p})\end{aligned}$$

Rodrigues formula



- Let us introduce a skew symmetric matrix

$$[\mathbf{u}]_\wedge = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix}$$

- Since

$$\begin{aligned} \mathbf{q} &= \mathbf{p} + \sin\theta(\mathbf{u} \wedge \mathbf{p}) \\ &\quad + (1 - \cos\theta)\mathbf{u} \wedge (\mathbf{u} \wedge \mathbf{p}) \end{aligned}$$

we have

$$\begin{aligned} \mathbf{q} &= \mathbf{R}\mathbf{p}, \\ \mathbf{R} &= \mathbf{I} + \sin\theta[\mathbf{u}]_\wedge + (1 - \cos\theta)[\mathbf{u}]_\wedge^2 \end{aligned}$$

- **Rodrigues rotation formula**

From rotation matrix to $(\theta, \mathbf{u})$

- Since $\mathbf{u}$ is a unit vector,

$$[\mathbf{u}]_\wedge^2 = \begin{bmatrix} u_x^2 - 1 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 - 1 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 - 1 \end{bmatrix}$$

- Take a trace of bothe hands of Rodrigues formula

$$\text{trace}\mathbf{R} = 3 + (1 - \cos\theta)(u_x^2 + u_y^2 + u_z^2 - 3) = 1 - 2\cos\theta$$

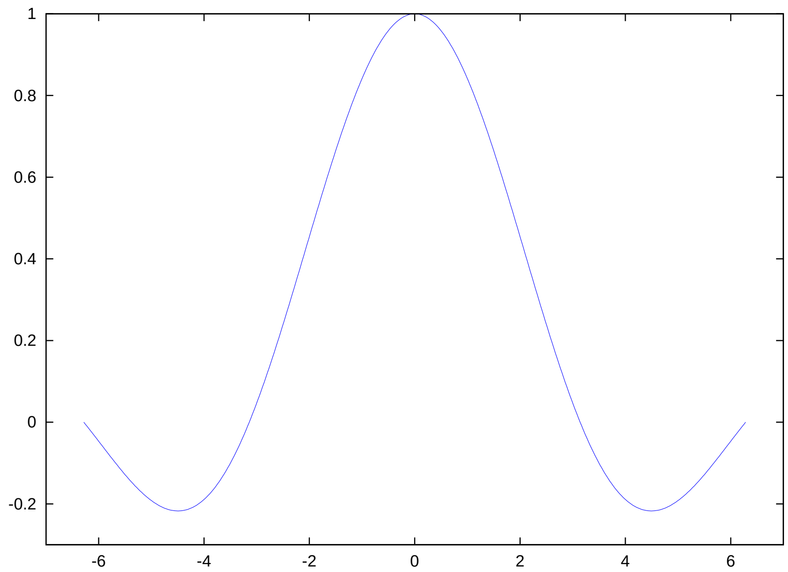
- Thus we have an equation for θ:

$$\theta = \arccos\left(\frac{1}{2}(r_{11} + r_{22} + r_{33} - 1)\right)$$

- On the other hand, since $\sin\theta = \theta\text{sinc}\theta$,

$$\mathbf{R} - \mathbf{R}^\top = 2\sin\theta([\mathbf{u}]_\wedge) = 2\text{sinc}\theta(\theta[\mathbf{u}]_\wedge)$$

From rotation matrix to $(\theta, \mathbf{u})$



- Picking up the off-diagonal elements of

$$\mathbf{R} - \mathbf{R}^\top = 2\text{sinc}\theta(\theta[\mathbf{u}]_\wedge)$$

yields

$$\mathbf{u}\theta = \frac{1}{2}\frac{1}{\text{sinc}\theta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

- This equation is singular only at $\theta = \pm\pi$. In this case $\mathbf{u}$ can be found as an eigenvector of $\mathbf{R}$ associated to the eigenvalue 1.

3D visual servo

- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

Angular velocity and rotation axis-angle

- When a vector $\mathbf{x}$ rotates with angular velocity $\boldsymbol{\omega}$, the velocity of the vector $\mathbf{x}$ is

$$\dot{\mathbf{x}} = \boldsymbol{\omega} \wedge \mathbf{x} = [\boldsymbol{\omega}]_\wedge \mathbf{x}$$

- Let the column vectors of $\mathbf{R}$ be $\mathbf{r}_x, \mathbf{r}_y, \mathbf{r}_z$, we have

$$\mathbf{R} = [\mathbf{r}_x \ \mathbf{r}_y \ \mathbf{r}_z], \quad \dot{\mathbf{r}}_x = [\boldsymbol{\omega}]_\wedge \mathbf{r}_x, \quad \dot{\mathbf{r}}_y = [\boldsymbol{\omega}]_\wedge \mathbf{r}_y, \quad \dot{\mathbf{r}}_z = [\boldsymbol{\omega}]_\wedge \mathbf{r}_z$$

- Thus we have

$$\dot{\mathbf{R}} = [\boldsymbol{\omega}]_\wedge \mathbf{R}$$

- And also we have

$$[\boldsymbol{\omega}]_\wedge = \dot{\mathbf{R}}\mathbf{R}^\top$$

Angular velocity and rotation axis-angle

- Derivative and transpose of Rodrigues formula

$$
\begin{aligned}
\dot{\mathbf{R}} &= \dot{\theta}\cos\theta[\mathbf{u}]_\wedge + \sin\theta[\dot{\mathbf{u}}]_\wedge \\
&\quad + \dot{\theta}\sin\theta[\mathbf{u}]_\wedge^2 + (1-\cos\theta)[\dot{\mathbf{u}}]_\wedge[\mathbf{u}]_\wedge \\
&\quad + (1-\cos\theta)[\mathbf{u}]_\wedge[\dot{\mathbf{u}}]_\wedge \\
\mathbf{R}^\top &= \mathbf{I} - \sin\theta[\mathbf{u}]_\wedge + (1-\cos\theta)[\mathbf{u}]_\wedge^2
\end{aligned}
$$

- Note that

$$
[\mathbf{u}]_\wedge^3 = -[\mathbf{u}]_\wedge, \quad [\mathbf{u}]_\wedge[\dot{\mathbf{u}}]_\wedge[\mathbf{u}]_\wedge = \mathbf{0}
$$

then we have

$$
\begin{aligned}
[\boldsymbol{\omega}]_\wedge &= \sin\theta[\dot{\mathbf{u}}]_\wedge + \dot{\theta}[\mathbf{u}]_\wedge \\
&\quad + (1-\cos\theta)[\mathbf{u}]_\wedge[\dot{\mathbf{u}}]_\wedge - (1-\cos\theta)[\dot{\mathbf{u}}]_\wedge[\mathbf{u}]_\wedge
\end{aligned}
$$

Angular velocity and rotation axis-angle

- Moreover, since

$$
\begin{aligned}
[\mathbf{u}]_\wedge[\mathbf{v}]_\wedge &= \mathbf{v}\mathbf{u}^\top - (\mathbf{u}^\top\mathbf{v})\mathbf{I}, \\
[\mathbf{u}]_\wedge[\mathbf{v}]_\wedge - [\mathbf{v}]_\wedge[\mathbf{u}]_\wedge &= \mathbf{v}\mathbf{u}^\top - \mathbf{u}\mathbf{v}^\top = [[\mathbf{u}]_\wedge\mathbf{v}]_\wedge
\end{aligned}
$$

  we have

$$[\boldsymbol{\omega}]_\wedge = \sin\theta[\dot{\mathbf{u}}]_\wedge + \dot{\theta}[\mathbf{u}]_\wedge + (1-\cos\theta)[[\mathbf{u}]_\wedge\dot{\mathbf{u}}]_\wedge$$

- By comparing both sides we have

$$\boldsymbol{\omega} = \dot{\theta}\dot{\mathbf{u}} + (\sin\theta\mathbf{I} + (1-\cos\theta)[\mathbf{u}]_\wedge)\dot{\mathbf{u}}$$

Angular velocity and rotation axis-angle

- Derivative of $\theta\mathbf{u}$:

$$\frac{d(\theta\mathbf{u})}{dt} = \dot{\theta}\mathbf{u} + \theta\dot{\mathbf{u}}$$

- Multiply $\mathbf{I} + [\mathbf{u}]_\wedge^2$ to both sides

$$\dot{\theta}\mathbf{u} = (\mathbf{I} + [\mathbf{u}]_\wedge^2)\frac{d(\theta\mathbf{u})}{dt}$$

- And multiply $-[\mathbf{u}]_\wedge^2$ to both sides

$$\theta\dot{\mathbf{u}} = -[\mathbf{u}]_\wedge^2\frac{d(\theta\mathbf{u})}{dt}$$

Angular velocity and rotation axis-angle

- On the other hand, since

$$1 - \cos\theta = \frac{\theta^2}{2}\text{sinc}^2\left(\frac{\theta}{2}\right), \quad \sin\theta = \theta\,\text{sinc}\,\theta$$

we have

$$\boldsymbol{\omega} = \dot{\theta}\dot{\mathbf{u}} + \left(\text{sinc}\theta\mathbf{I} + \frac{\theta}{2}\text{sinc}^2\left(\frac{\theta}{2}\right)[\mathbf{u}]_\wedge\right)\theta\dot{\mathbf{u}}$$

- Finally

$$\boldsymbol{\omega} = \left(\mathbf{I} + \frac{\theta}{2}\text{sinc}^2\left(\frac{\theta}{2}\right)[\mathbf{u}]_\wedge + (1 - \text{sinc}\,\theta)[\mathbf{u}]_\wedge^2\right)\frac{d(\theta\mathbf{u})}{dt}$$

- By computing the inverse of the matrix on the right hand side, we have

$$\frac{d(\theta\mathbf{u})}{dt} = \mathbf{J}_{\theta\mathbf{u}}\boldsymbol{\omega}, \quad \mathbf{J}_{\theta\mathbf{u}} = \mathbf{I} - \frac{\theta}{2}[\mathbf{u}]_\wedge + \left(1 - \frac{\text{sinc}\theta}{\text{sinc}^2\frac{\theta}{2}}\right)[\mathbf{u}]_\wedge^2$$

Position-based control

- From the image output $\mathbf{m}$, compute the controlled value $\mathbf{s}$. And compare $\mathbf{s}$ with the desired value $\mathbf{s}^*$. Derive the input $\mathbf{v}$ so that $\mathbf{s}$ converges to $\mathbf{s}^*$.
- In position-based visual servo, $\mathbf{s}$ is selected as a 3D parameter.
- The controlled error is

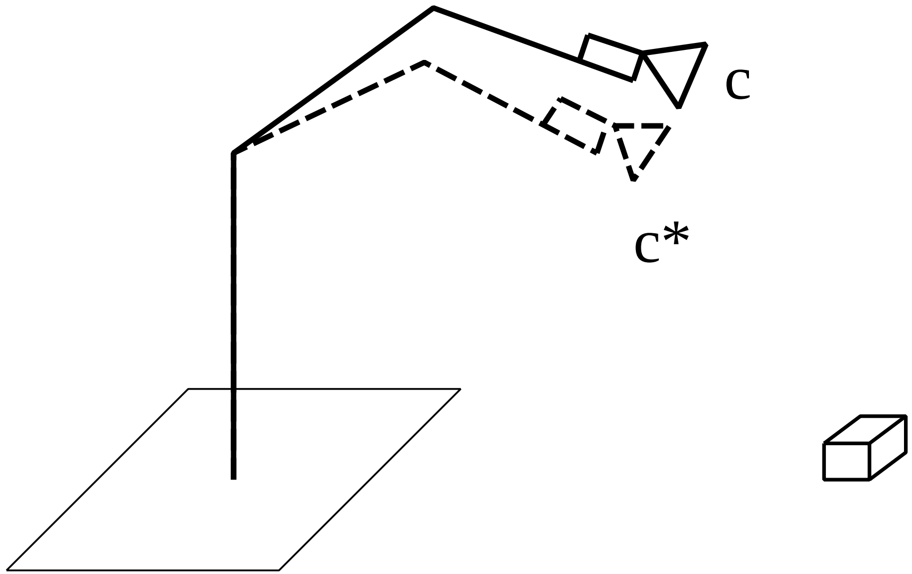
$$\mathbf{e}(t) = \mathbf{s}(\mathbf{m}(t), \mathbf{v}(t), \mathbf{a}) - \mathbf{s}^*$$

where $\mathbf{a}$ includes all parameters such as intrinsic and extrinsic parameters of the camera, object shape and size.

3D visual servo

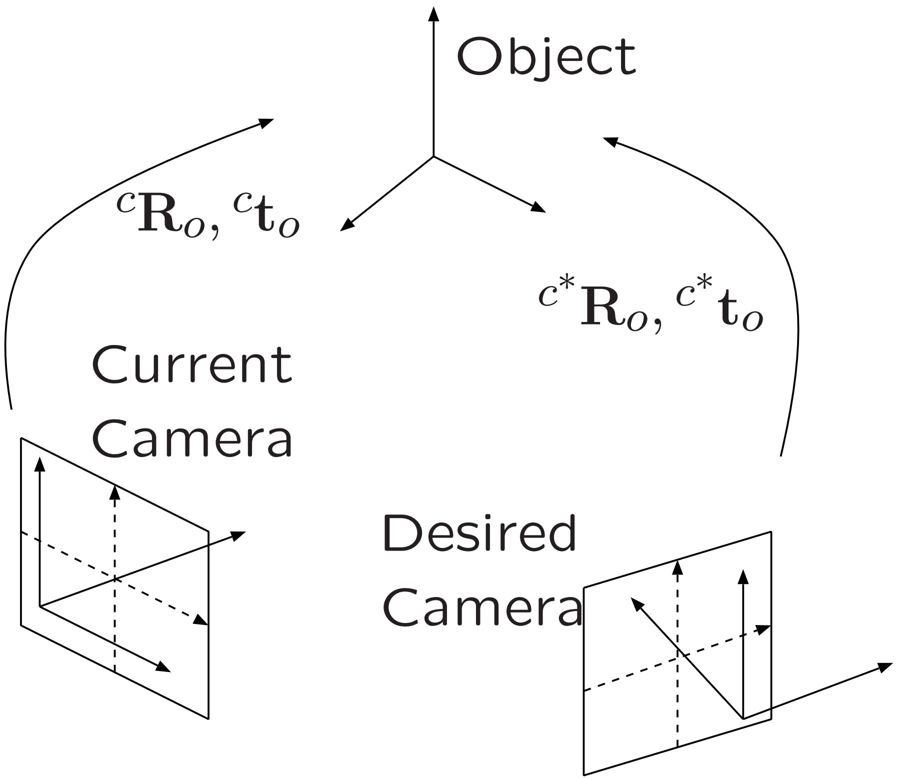
- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

Position-based control



- For example, a camera is mounted on the robot hand and we want to control the camera position and orientation $\mathbf{c}$ to the desired position and orientation $\mathbf{c}^*$.
- Note that the relationship between the object and the camera is not explicitly controlled. Only the relationship between $\mathbf{c}$ and $\mathbf{c}^*$ is important.

Position-based control



- Homography-based algorithm can be used to find $\mathbf{R}, \mathbf{t}$.
- object - camera: $^{c}\mathbf{t}_o$
- object - desired camera: $^{c^*}\mathbf{t}_o$

Control law

- Controlled variables:

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^*, \quad \mathbf{s} = \begin{bmatrix} {}^c\mathbf{t}_o \\ \theta\mathbf{u} \end{bmatrix}, \quad \mathbf{s}^* = \begin{bmatrix} {}^{c^*}\mathbf{t}_o \\ \mathbf{0} \end{bmatrix}$$

- Control input:

$$\mathbf{v} = \begin{bmatrix} {}^c\mathbf{v}_c \\ {}^c\boldsymbol{\omega}_c \end{bmatrix}$$

Control law

- Relationship between them:

$$\frac{d^c\mathbf{t}_o}{dt} = -^c\mathbf{v}_c - {}^c\boldsymbol{\omega}_c \wedge {}^c\mathbf{t}_o = -^c\mathbf{v}_c + [^c\mathbf{t}_o]_\wedge {}^c\boldsymbol{\omega}_c$$

$$\frac{d(\theta\mathbf{u})}{dt} = \mathbf{J}_{\theta\mathbf{u}}\boldsymbol{\omega}, \quad \mathbf{J}_{\theta\mathbf{u}} = \mathbf{I} - \frac{\theta}{2}[\mathbf{u}]_\wedge + \left(1 - \frac{\mathrm{sinc}\theta}{\mathrm{sinc}^2\frac{\theta}{2}}\right)[\mathbf{u}]_\wedge^2$$

- Thus we have

$$\dot{\mathrm{e}} = \mathbf{J}\mathbf{v}, \quad \mathbf{J} = \begin{bmatrix} -\mathbf{I} & [^c\mathbf{t}_o]_\wedge \\ \mathbf{0} & \mathbf{J}_{\theta\mathbf{u}} \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} {}^c\mathbf{v}_c \\ {}^c\boldsymbol{\omega}_c \end{bmatrix}$$

Control law

- The control law

$$\mathbf{v} = -\lambda\ \mathbf{J}^{-1}\mathbf{e} = -\lambda \begin{bmatrix} {}^{c}\mathbf{t}_o - {}^{c^*}\mathbf{t}_o + [{}^{c}\mathbf{t}_o]_\wedge \theta\mathbf{u} \\ \theta\mathbf{u} \end{bmatrix}$$

- The closed loop system

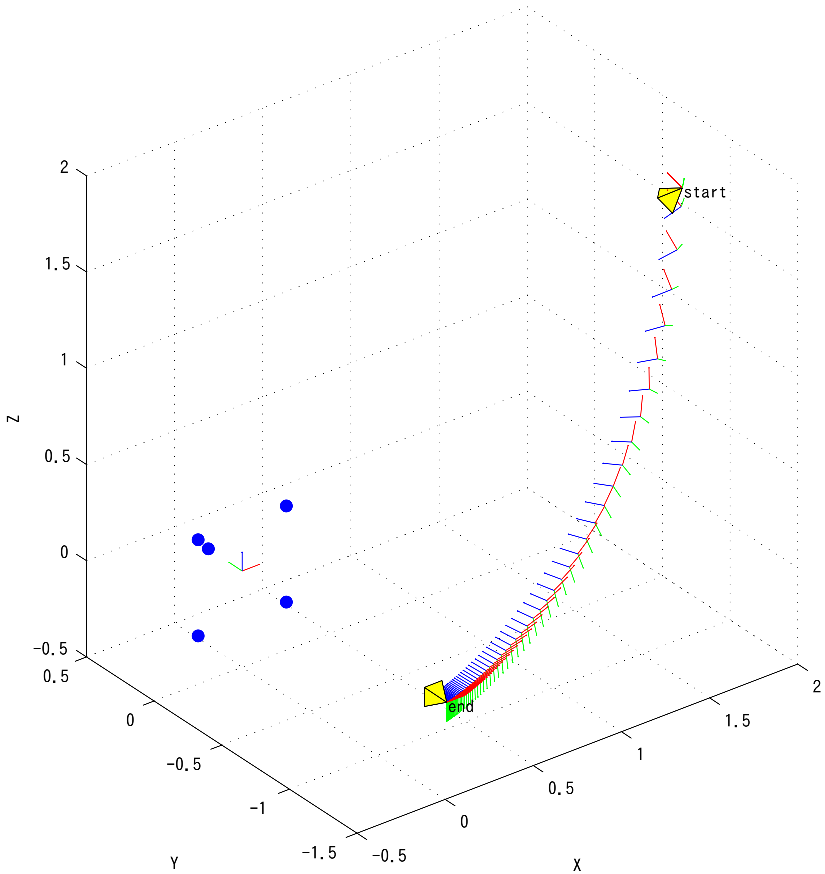
$$\dot{\mathbf{e}} = -\lambda\mathbf{e}, \quad \mathbf{e}(t) = e^{-\lambda t}\mathbf{e}_0$$

- $\mathbf{J}_{\theta\mathbf{u}}$ becomes singular at $\theta = \pm 2\pi$.
- Near the origin $\theta = 0$, $\mathbf{J}_{\theta\mathbf{u}} \approx \mathbf{I}$.
- The system well behaves for practically important region.
- The stability region is almost global.
- This control law do not care how the image will change.
- Indeed, no guarantee to keep the object in the field of view.

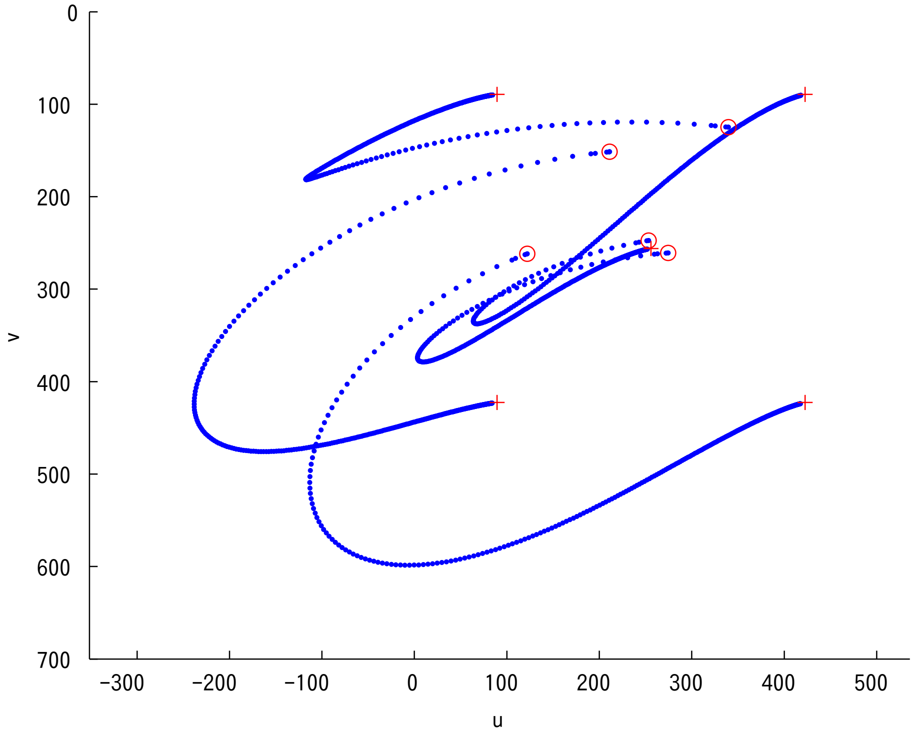
Simulation

```
T0c_0 =

   -0.46742     0.67153    -0.57495      1.6598
    0.49873     0.73729      0.4557       -1.09
    0.72992   -0.073743    -0.67954      1.9059
          0           0           0           1

T0c_x =

          1           0           0           0
          0           0           1        -1.5
          0          -1           0           0
          0           0           0           1
```

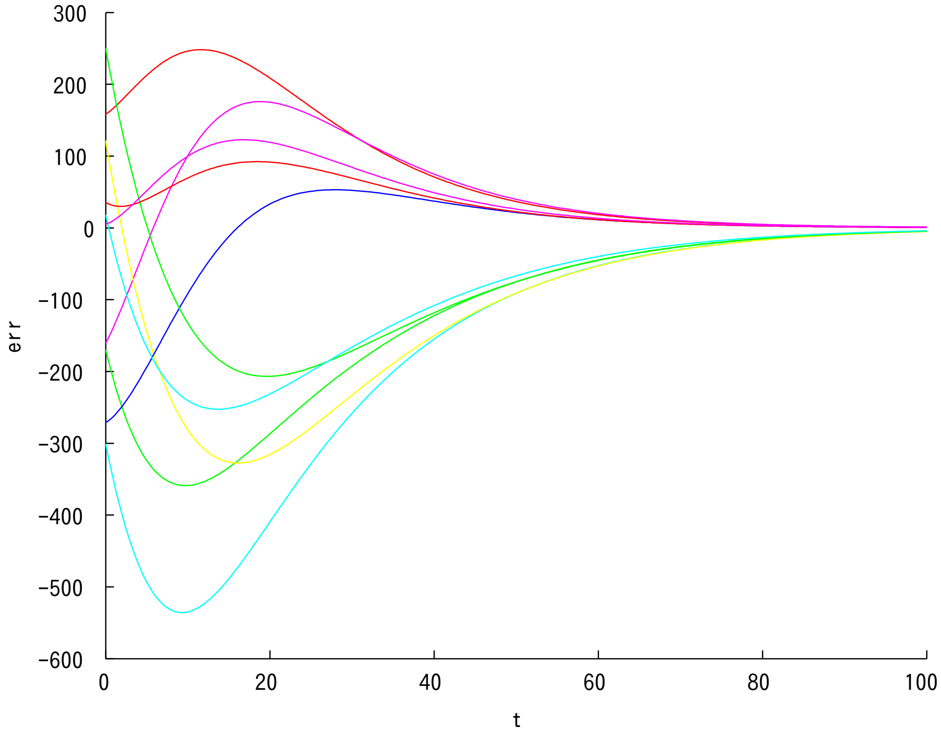
Camera trajectory



Feature points trajectory



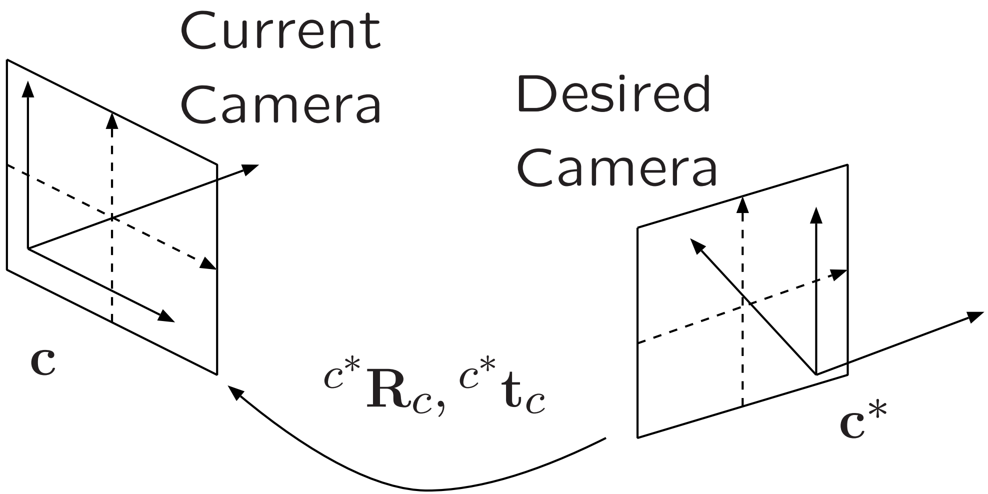
Trajectory error



3D visual servo

- History
- Expression of rotation
- Expression of angular velocity
- Position-based visual servo I
- Position-based visual servo II

Position-based control: Another choice



- **Controlled variable**: Relative position expressed in the desired coordinate system.

$$\mathbf{t} = {}^{c^*}\mathbf{t}_c = {}^{c^*}\mathbf{c}^* - {}^{c^*}\mathbf{c}, \quad \mathbf{R} = {}^{c^*}\mathbf{R}_c$$

- Express orientation error using $\theta\mathbf{u}$

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^*, \quad \mathbf{s} = \left[\begin{array}{c} \mathbf{t} \\ \theta\mathbf{u} \end{array} \right], \quad \mathbf{s}^* = \mathbf{0}$$

Control law

- Control input:

$$\mathbf{v} = \begin{bmatrix} {}^c\mathbf{v}_c \\ {}^c\boldsymbol{\omega}_c \end{bmatrix}$$

- Relationship between input and controlled variables

$$\begin{aligned} \frac{d\mathbf{t}}{dt} &= \frac{d}{dt}\left(-{}^{c^*}\mathbf{R}_c{}^c\mathbf{c}\right) = -\dot{\mathbf{R}}{}^c\mathbf{c} - \mathbf{R}{}^c\dot{\mathbf{c}} \\ &= [{}^c\boldsymbol{\omega}_c]_\wedge \mathbf{R}{}^c\mathbf{c} - \mathbf{R}\left(-{}^c\mathbf{v}_c - [{}^c\boldsymbol{\omega}_c]_\wedge{}^c\mathbf{c}\right) \;=\; \mathbf{R}{}^c\mathbf{v}_c \end{aligned}$$

- Thus we have

$$\dot{\mathrm{e}} = \dot{\mathrm{s}} = \mathbf{J}\mathbf{v}, \quad \mathbf{J} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\theta\mathbf{u}} \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} {}^c\mathbf{v}_c \\ {}^c\boldsymbol{\omega}_c \end{bmatrix}$$

Control law

- Control law

$$\mathbf{v} = -\lambda \mathbf{J}^{-1}\mathbf{e} = -\lambda \begin{bmatrix} \mathbf{R}^\top \mathbf{t} \\ \theta \mathbf{u} \end{bmatrix}$$

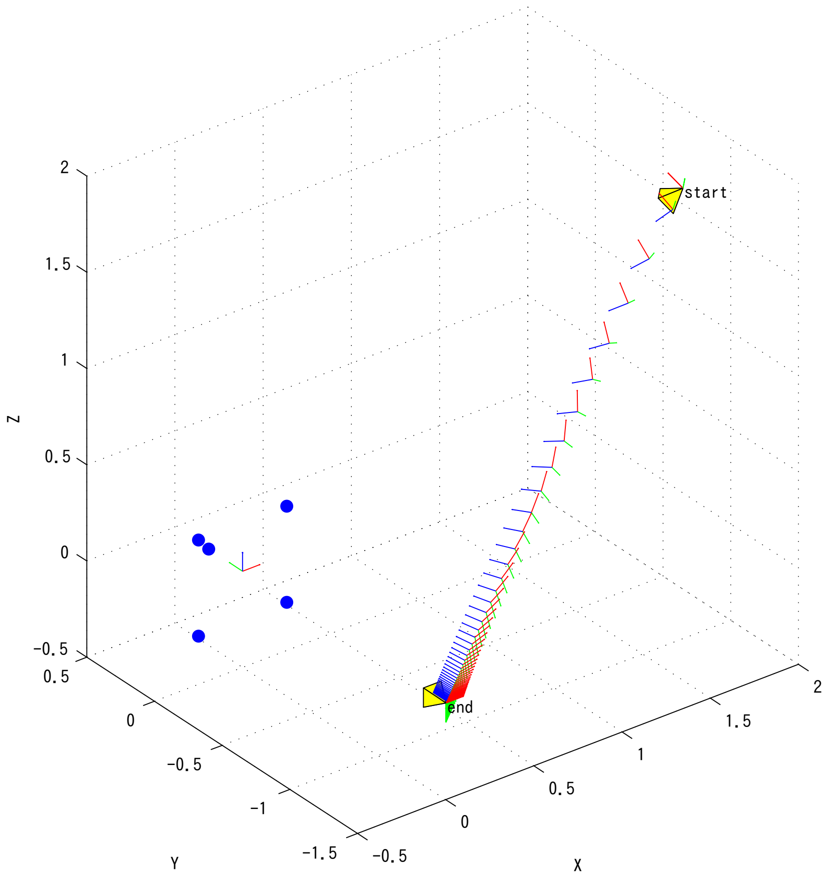
- Closed loop system

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \quad \mathbf{e}(t) = e^{-\lambda t}\mathbf{e}_0$$

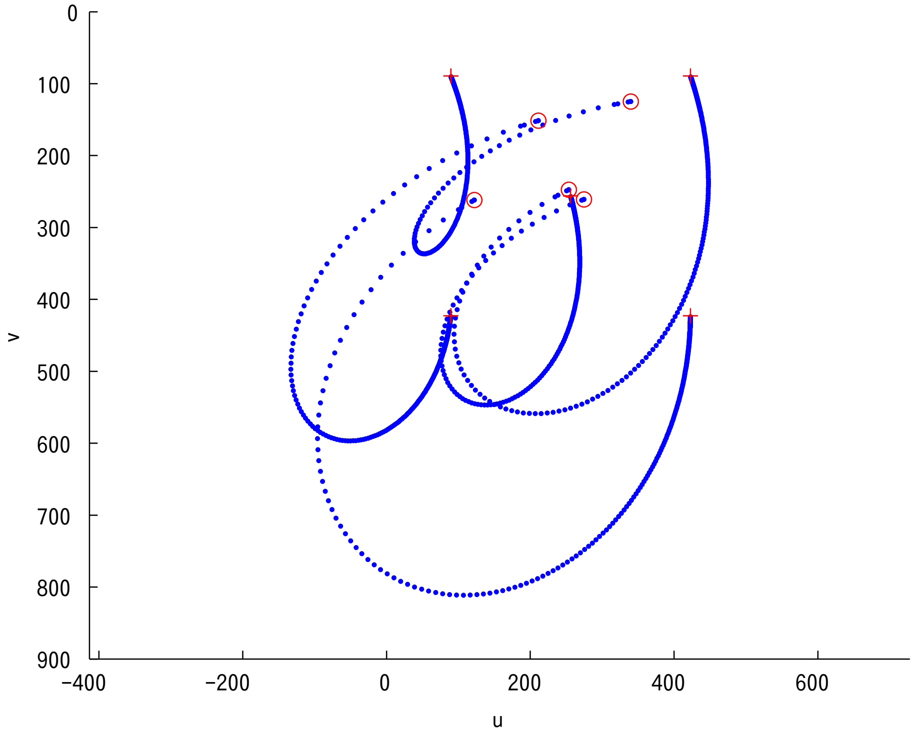
- $\mathbf{J}_{\theta\mathbf{u}}$ becomes singular at $\theta = \pm 2\pi$.
- Near the origin $\theta = 0$, $\mathbf{J}_{\theta\mathbf{u}} \approx \mathbf{I}$.
- The system well behaves for practically important region.
- The stability region is almost global.
- This control law do not care how the image will change.
- Indeed, no guarantee to keep the object in the field of view.

```
T0c_0 =

   -0.46742     0.67153    -0.57495     1.6598
    0.49873     0.73729     0.4557     -1.09
    0.72992   -0.073743   -0.67954     1.9059
          0           0           0          1

T0c_x =

          1           0           0          0
          0           0           1       -1.5
          0          -1           0          0
          0           0           0          1
```

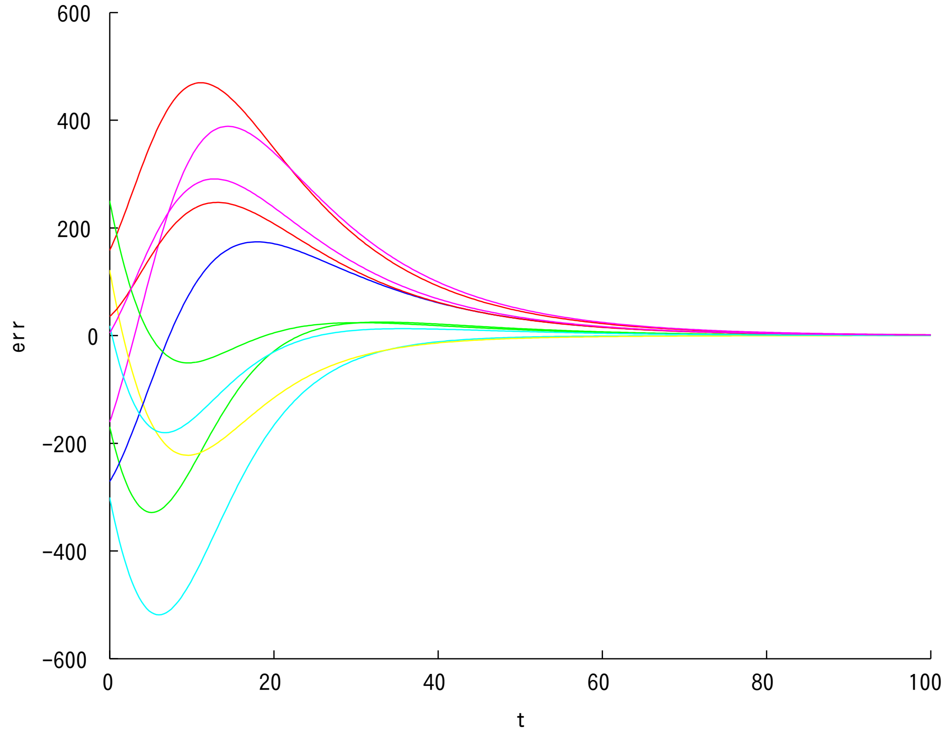
Camera trajectory



Feature points trajectory



Trajectory error



Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- Sampling time issues
- ESM algorithm and visual tracking

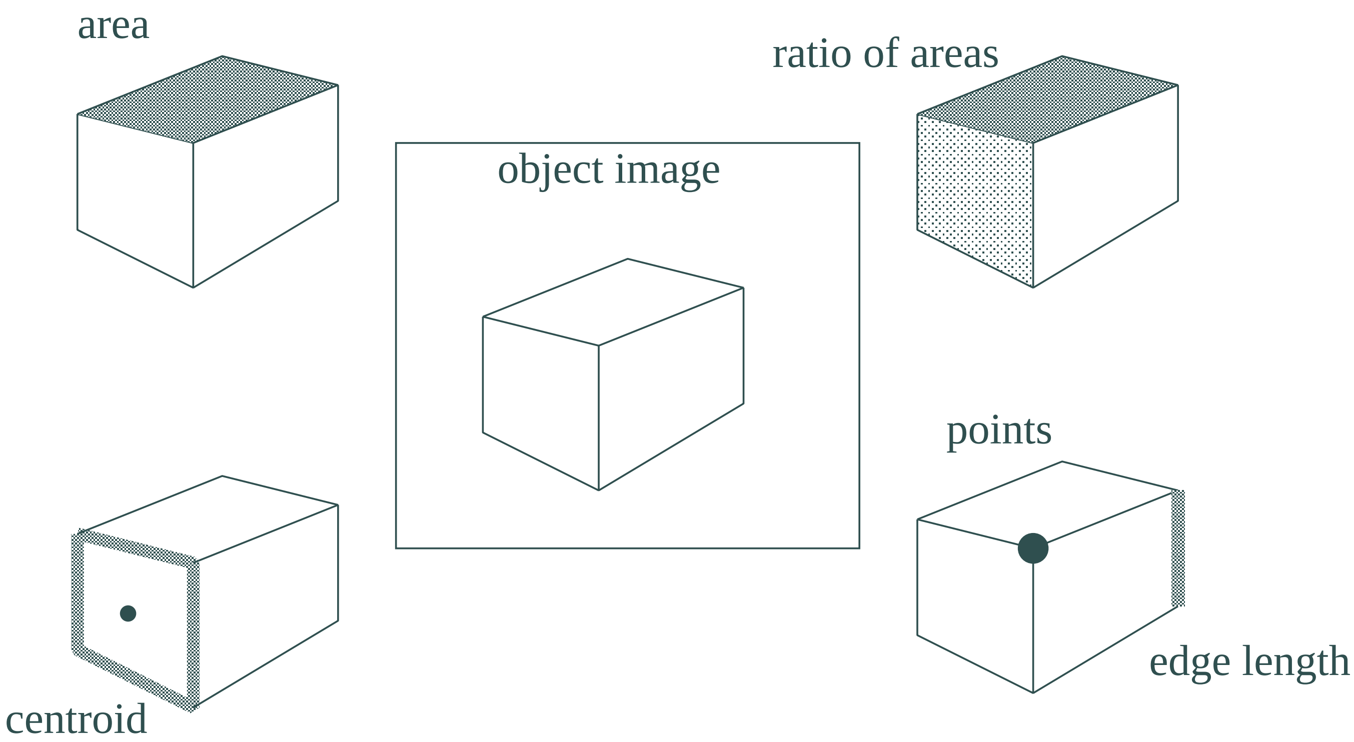
2D visual servo

- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

2D visual servo

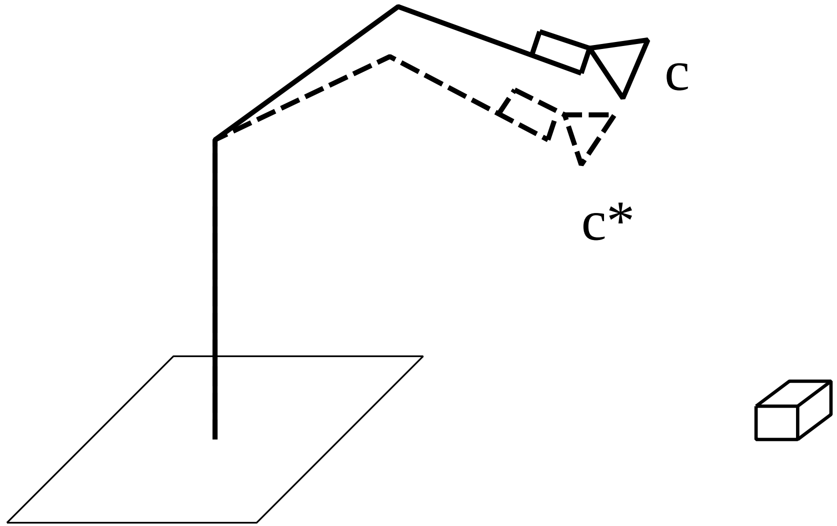
- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

Image features



- **Image features**: easy to extract, must change if camera position changes, number of features must be larger than the number of robot DOF

Visual servo



- Image output: $\mathbf{m}$
- Controlled variables: $\mathbf{s}$
- Desired value: $\mathbf{s}^*$
- Control input: $\mathbf{v}$
- Design a controller so that: $\mathbf{s} \rightarrow \mathbf{s}^*$

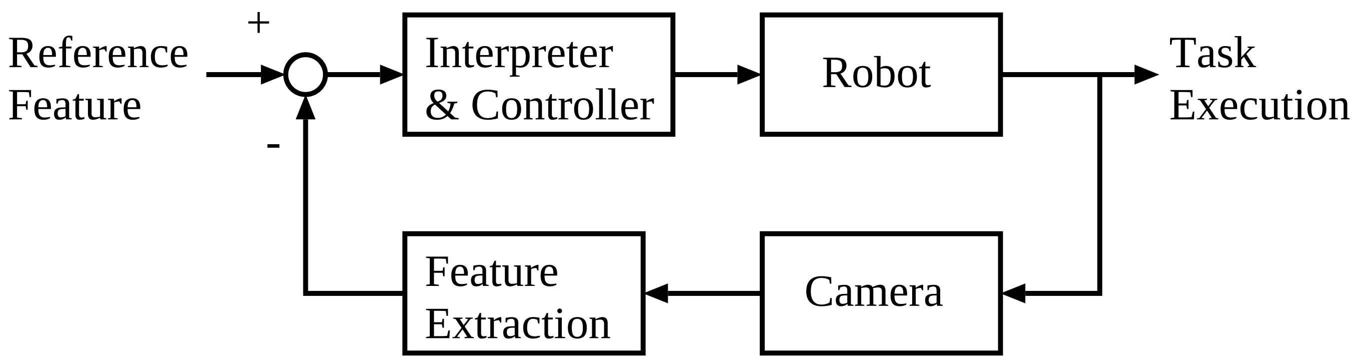
Feature-based visual servo formulation

- Controlled variables $\mathbf{s}$ as image features
- Desired value $\mathbf{s}^*$ as desired value of image features
- Find $\mathbf{v}$ so that

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{m}(t), \mathbf{v}(t), \mathbf{a}) - \mathbf{s}^*$$

is minimized. Here $\mathbf{a}$ includes intrinsic and extrinsic parameters of the camera.

Feature-based visual servo formulation



- This scheme does not require complicated object pose estimation.
- Parameters on the object shape and size are not required.
- Desired features are generated by teach-by-showing.

2D visual servo

- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

Image Jacobi matrix

- Use points image as features.
- Point coordinate in 3D

$$\mathbf{p} = [X \ Y \ Z]^\top$$

- The feature coordinate is

$$\mathbf{x} = [x \ y]^\top = [X/Z \ Y/Z]^\top$$

- Control input: robot hand position and orientation $\mathbf{q}$
- **Image Jacobi matrix**:

$$\mathbf{J}_\mathbf{x} = \frac{\partial \mathbf{x}}{\partial \mathbf{q}}$$

Image Jacobi matrix

- Input velocity: $\mathbf{v} = \dot{\mathbf{q}}$
- Output velocity: $\dot{\mathbf{x}}$
- Image Jacobi matrix: $\dot{\mathbf{x}} = \mathbf{J}_{\mathbf{x}}\mathbf{v}$
- Derivative of $\mathbf{x}$

$$
\begin{aligned}
\dot{x} &= \frac{d}{dt}\left(\frac{X}{Z}\right) = \frac{\dot{X}Z - X\dot{Z}}{Z^2} \\
\dot{y} &= \frac{d}{dt}\left(\frac{Y}{Z}\right) = \frac{\dot{Y}Z - Y\dot{Z}}{Z^2}
\end{aligned}
$$

Image Jacobi matrix

- Point velocity due to camera motion

$$\dot{\mathbf{p}} = -\mathbf{v}_c - \boldsymbol{\omega}_c \wedge \mathbf{p}$$

- Elements of velocity and angular velocity

$$\mathbf{v}_c = [v_x \ v_y \ v_z]^\top, \quad \boldsymbol{\omega}_c = [\omega_x \ \omega_y \ \omega_z]^\top$$

- Then we have

$$\begin{aligned}
\dot{X} &= -v_x - \omega_y Z + \omega_z Y \\
\dot{Y} &= -v_y - \omega_z X + \omega_x Z \\
\dot{Z} &= -v_z - \omega_x Y + \omega_y X
\end{aligned}$$

- Substituting these equations into image velocity

$$\begin{aligned}
\dot{x} &= -v_x/Z + x v_z/Z + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\
\dot{y} &= -v_y/Z + y v_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z
\end{aligned}$$

Image Jacobi matrix

- In summary we have

$$
\begin{aligned}
\dot{\mathbf{x}} &= \mathbf{J}_{\mathbf{x}}\mathbf{v} \\
\mathbf{J}_{\mathbf{x}} &= \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix} \\
\mathbf{v} &= \begin{bmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{bmatrix}
\end{aligned}
$$

where Z is the depth.

- Stacking this relationship for n points yields

$$
\mathbf{e} = \mathbf{s} - \mathbf{s}^*, \quad \mathbf{s} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{s}^* = \begin{bmatrix} \mathbf{x}_1^* \\ \mathbf{x}_2^* \\ \vdots \\ \mathbf{x}_n^* \end{bmatrix}
$$

2D visual servo

- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

System description

- System description

$$\dot{\mathbf{e}} = \mathbf{J}\mathbf{v}$$

  where

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_{\mathbf{x}_1} \\ \mathbf{J}_{\mathbf{x}_2} \\ \vdots \\ \mathbf{J}_{\mathbf{x}_n} \end{bmatrix}$$

  is called the image Jacobi matrix.

- Number of features n should be equal to or larger than the robot DOF m. In this case the image Jacobi matrix $\mathbf{J} \in \mathbb{R}^{n \times m}$ becomes tall.

Control law

- A control law is given by

$$\mathbf{v} = -\lambda \mathbf{J}^{\dagger}\mathbf{e}, \quad \mathbf{J}^{\dagger} = (\mathbf{J}^{\top}\mathbf{J})^{-1}\mathbf{J}^{\top}, \quad \mathbf{e} = \mathbf{s} - \mathbf{s}^{*}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{v}_C \\ \boldsymbol{\omega}_C \end{bmatrix}$$

- In this case the error dynamics becomes

$$\dot{\mathbf{e}} = \mathbf{J}^{*}\mathbf{v} = -\lambda \mathbf{J}(\mathbf{J}^{\top}\mathbf{J})^{-1}\mathbf{J}^{\top}\mathbf{e}$$

- Note that

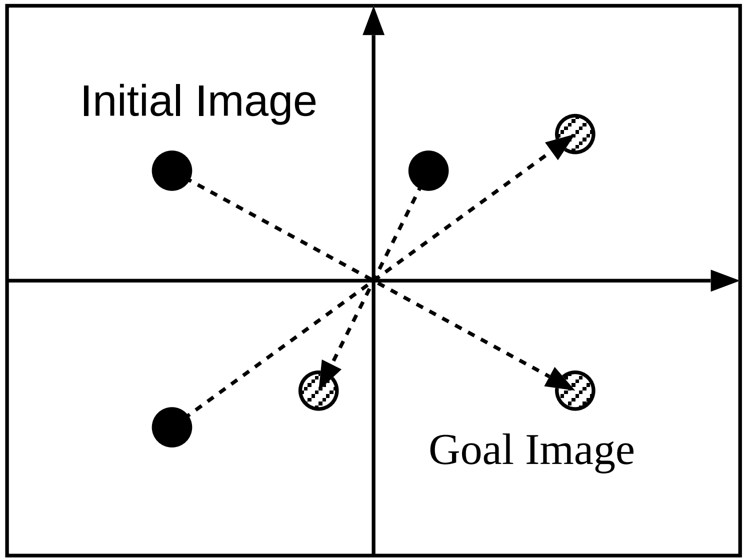
$$\mathbf{J}(\mathbf{J}^{\top}\mathbf{J})^{-1}\mathbf{J}^{\top} \neq \mathbf{I}$$

so it requires more discussion.

2D visual servo

- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

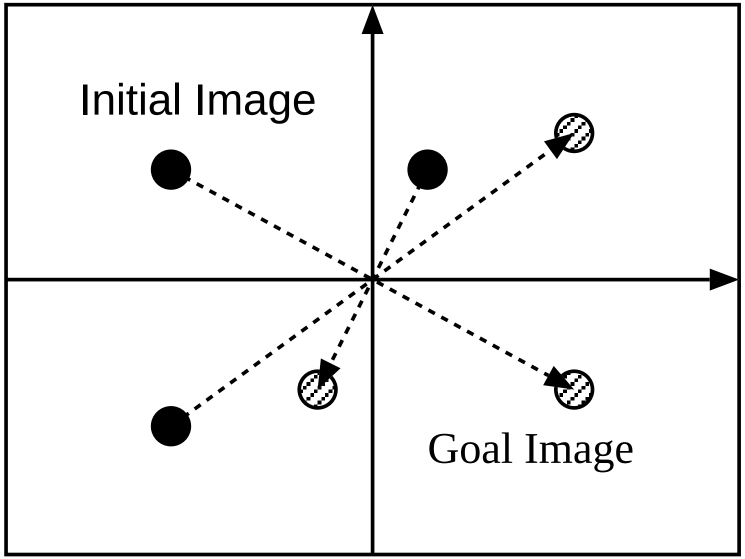
Undesired motion



- Suppose that initial and desired features are at 180 degree rotated around the image center.
- Consider a control law

$$\mathbf{v} = -\lambda \mathbf{J}^{\dagger}(\mathbf{s} - \mathbf{s}^{*})$$

Undesired motion



- This control law yields a motion that the feature points move straightly to the desired points.
- Then the object image becomes infinitely small at the image center.
- This means that the camera moves infinitely far away from the object.

2D visual servo

- Features and formulation
- Image Jacobi matrix
- Control law
- Undesired motion
- Simulation

Selection of control gain matrix

- Generalized inverse

$$\mathbf{v} = -\lambda\ \mathbf{J}^{\dagger}(\boldsymbol{\theta})(\mathbf{s} - \mathbf{s}^*)$$

- Fixed gain

$$\mathbf{v} = -\lambda\ \mathbf{J}^{*\dagger}(\mathbf{s} - \mathbf{s}^*)$$

where $\mathbf{J}^* = \mathbf{J}(\boldsymbol{\theta}^*)$

- ESM

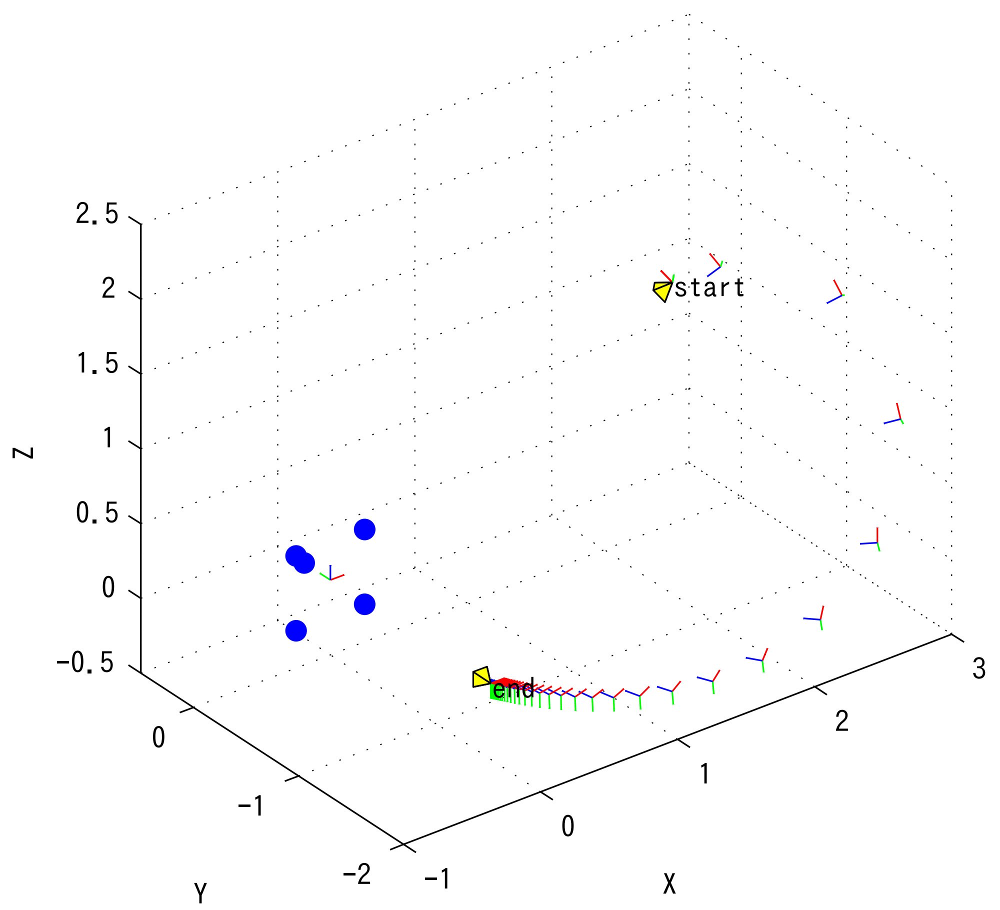
$$\mathbf{v} = -\lambda\ \mathbf{J}\mathsf{esm}^{\dagger}(\boldsymbol{\theta})(\mathbf{s} - \mathbf{s}^*)$$

where

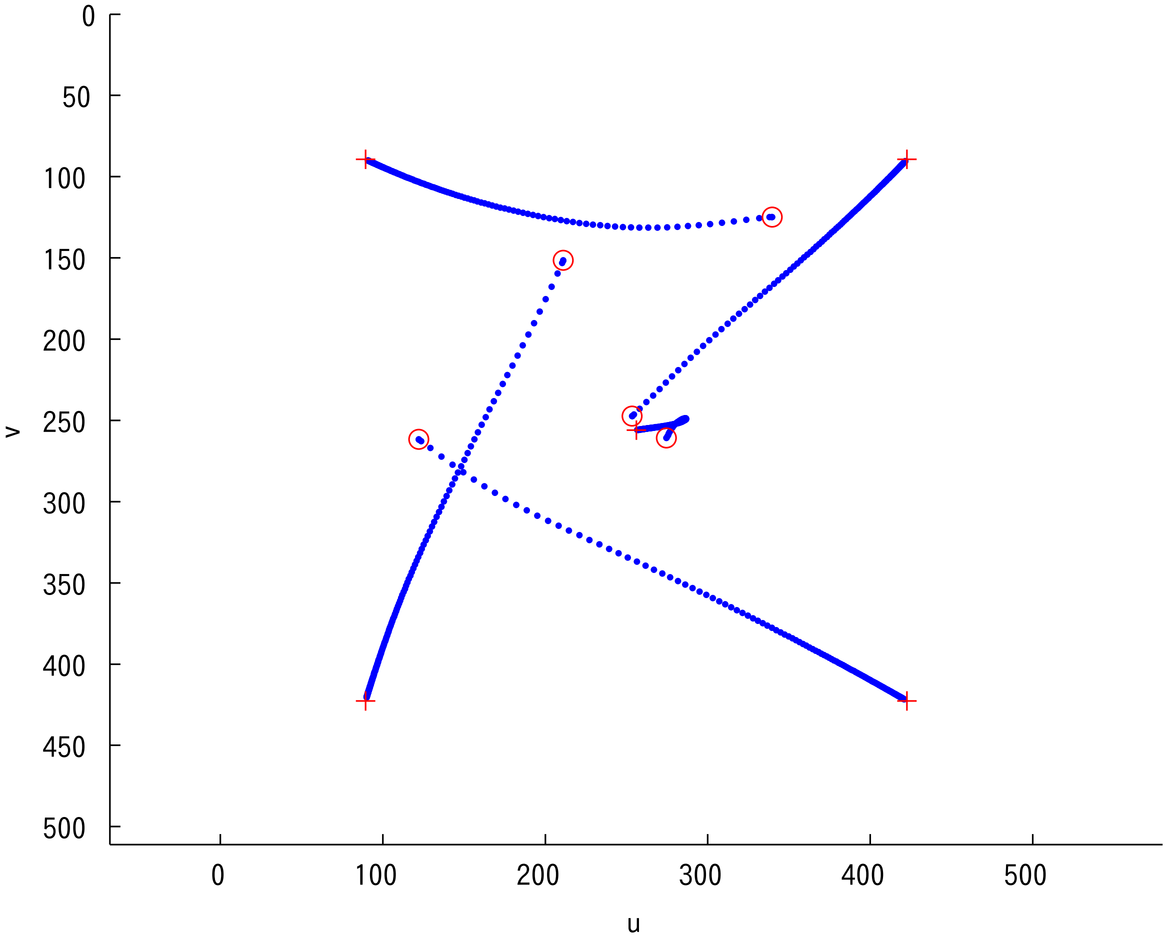
$$\mathbf{J}_{\mathsf{esm}}(\boldsymbol{\theta}) = (\mathbf{J}(\boldsymbol{\theta}) + \mathbf{J}^*)/2$$

```
T0c_0 =

   -0.46742     0.67153    -0.57495     1.6598
    0.49873     0.73729     0.4557     -1.09
    0.72992    -0.073743   -0.67954     1.9059
          0           0           0          1

T0c_x =

          1           0           0          0
          0           0           1       -1.5
          0          -1           0          0
          0           0           0          1
```

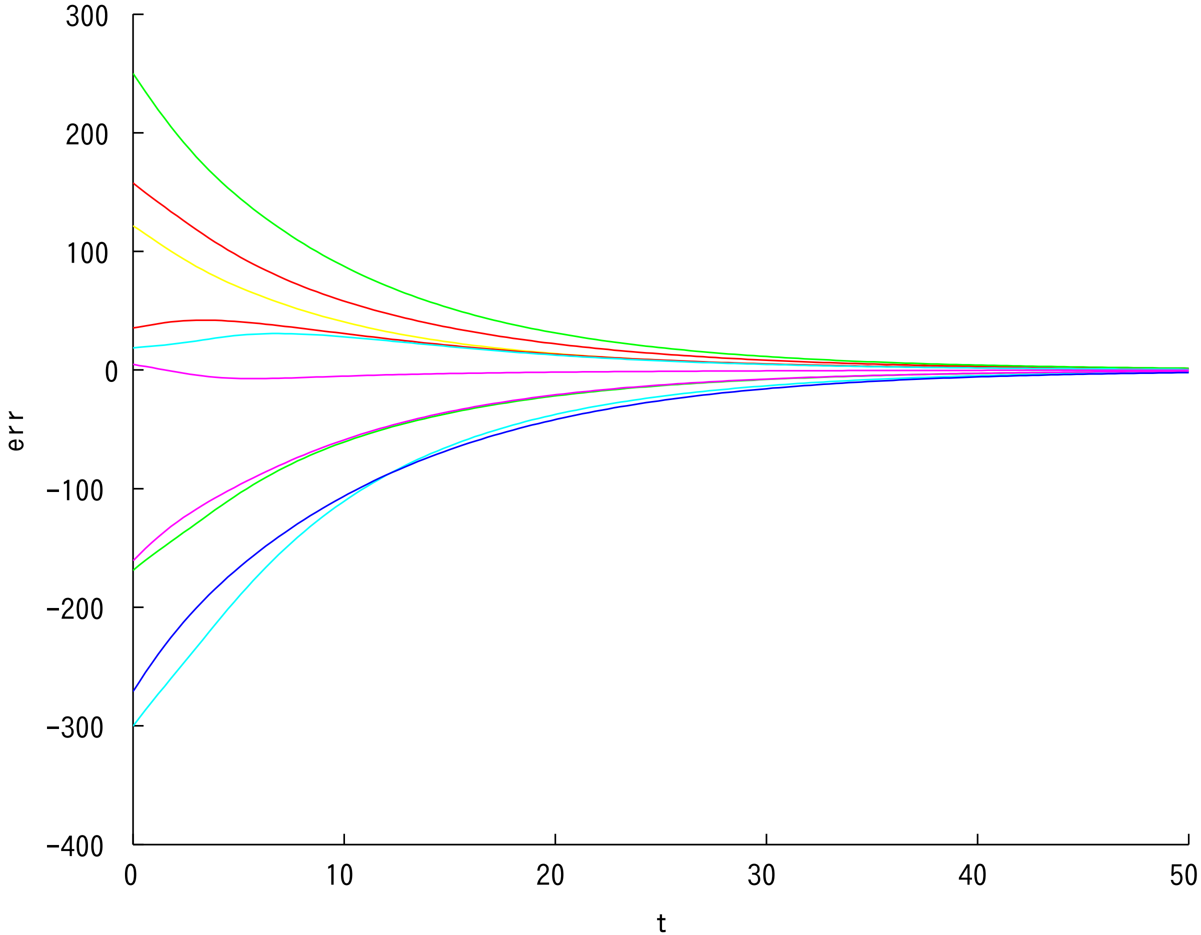
Generalized inverse: Camera trajectory



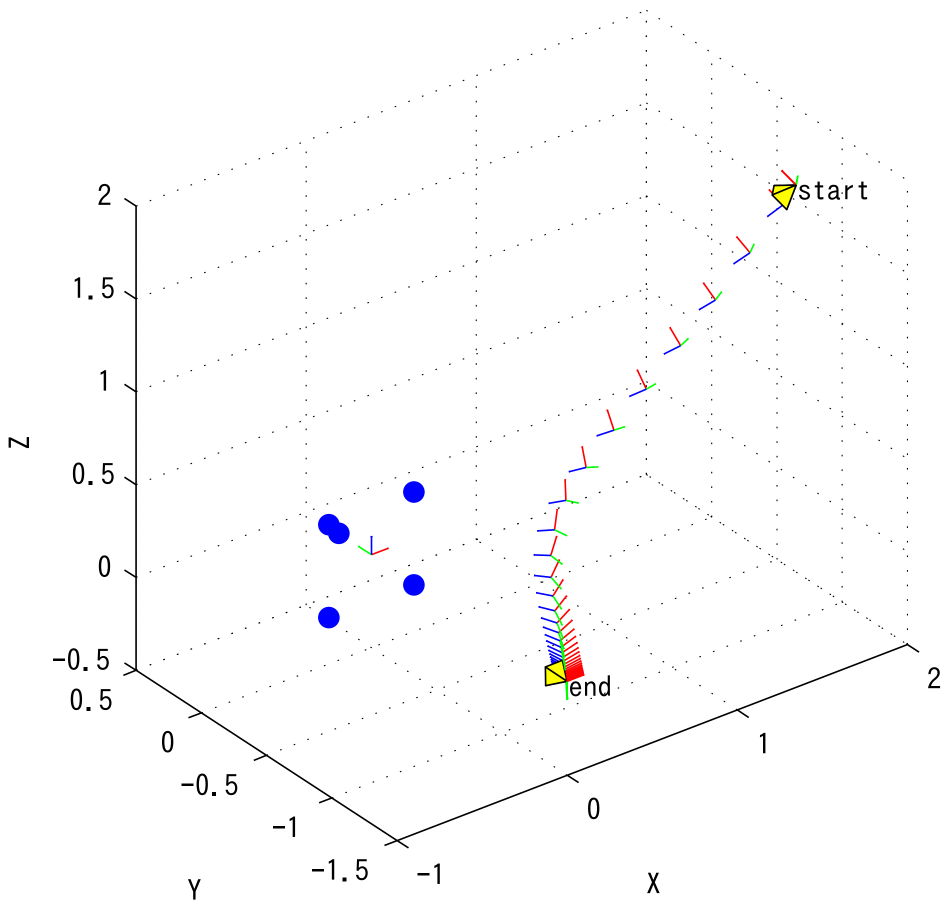
Generalized inverse: Feature trajectory



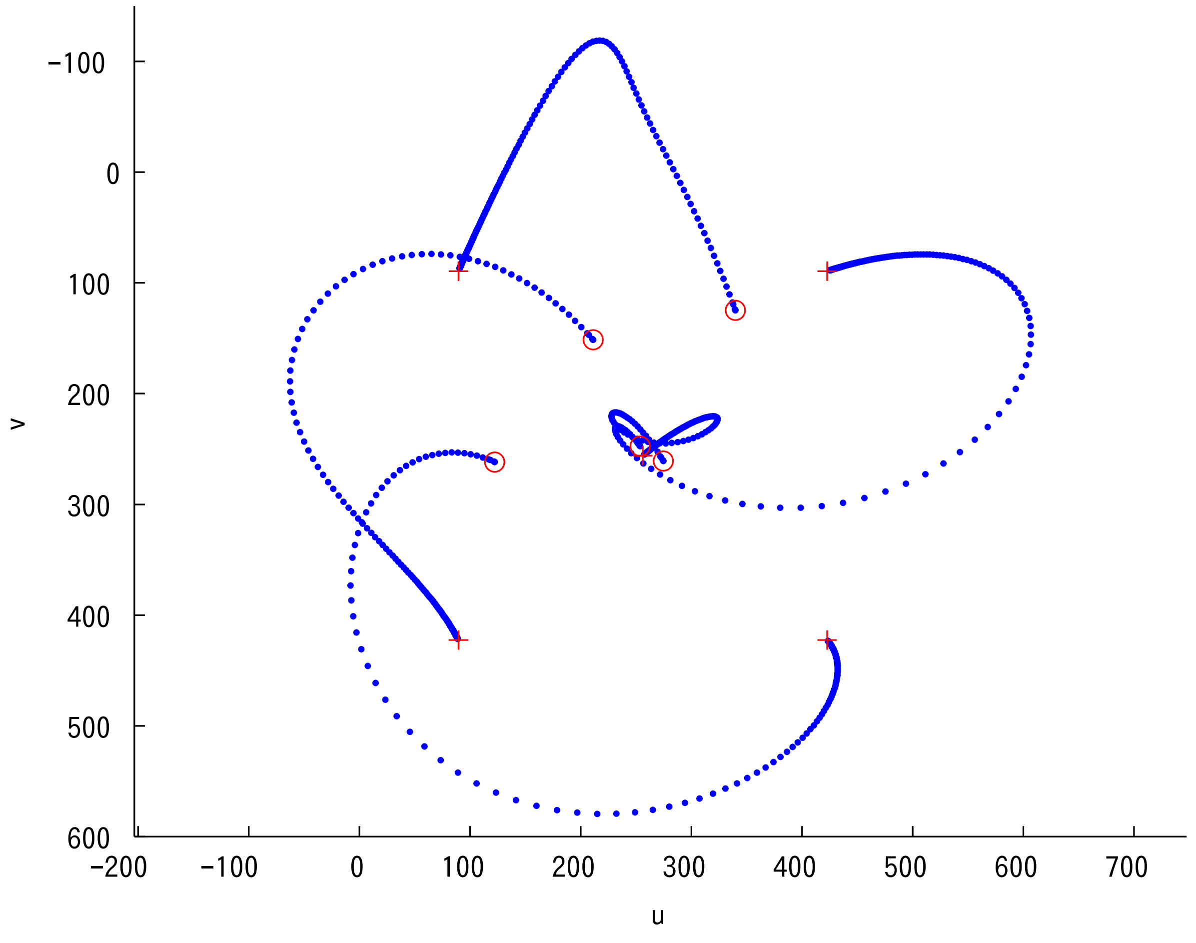
Generalized inverse: Feature error



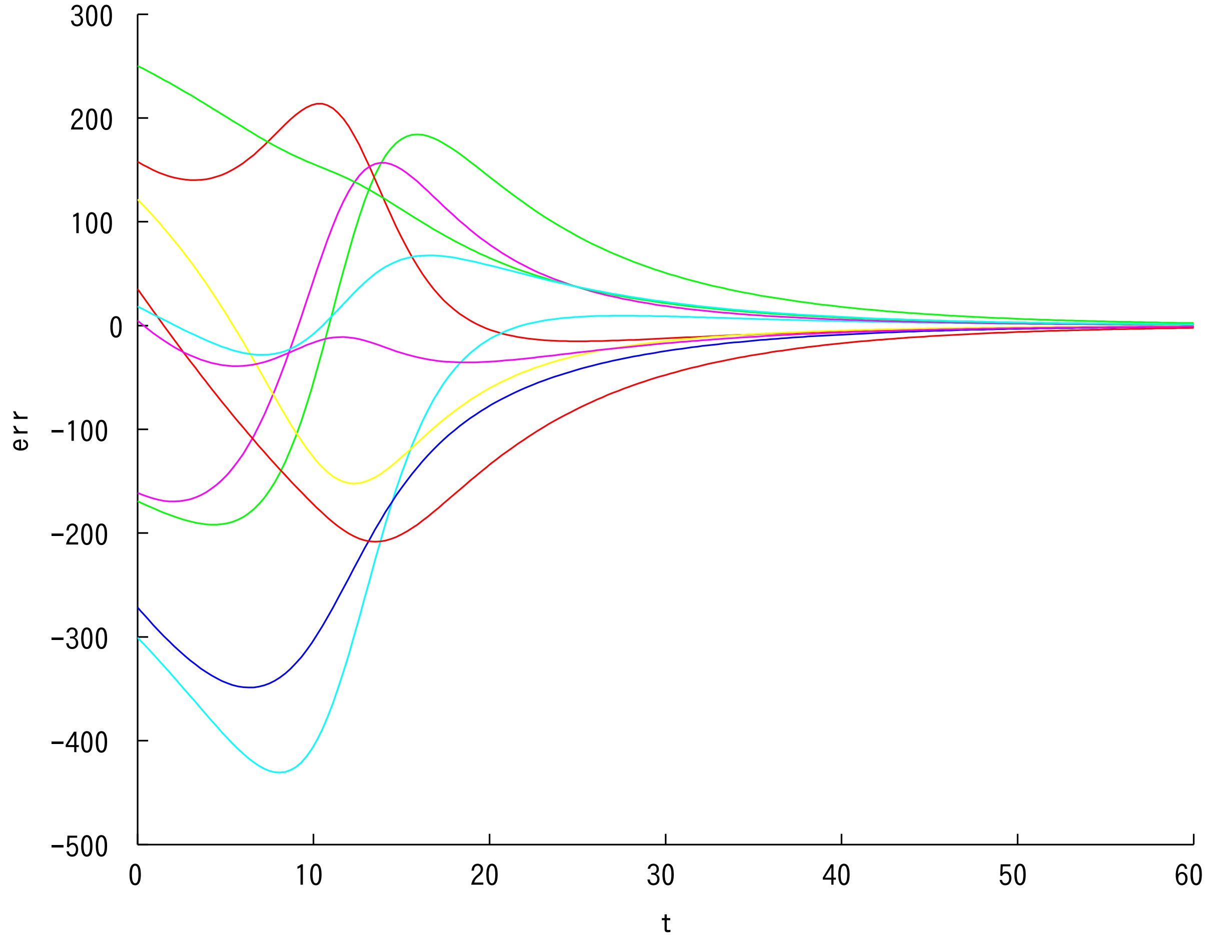
Fixed gain: Camera trajectory



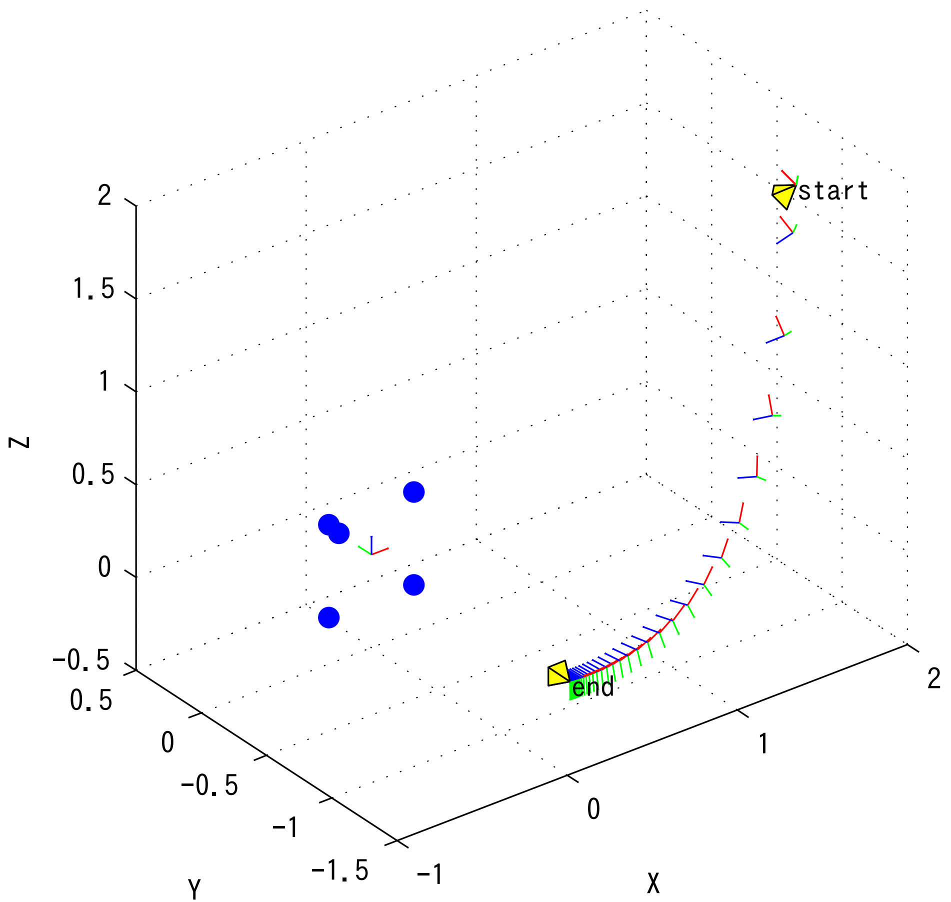
Fixed gain: Feature trajectory



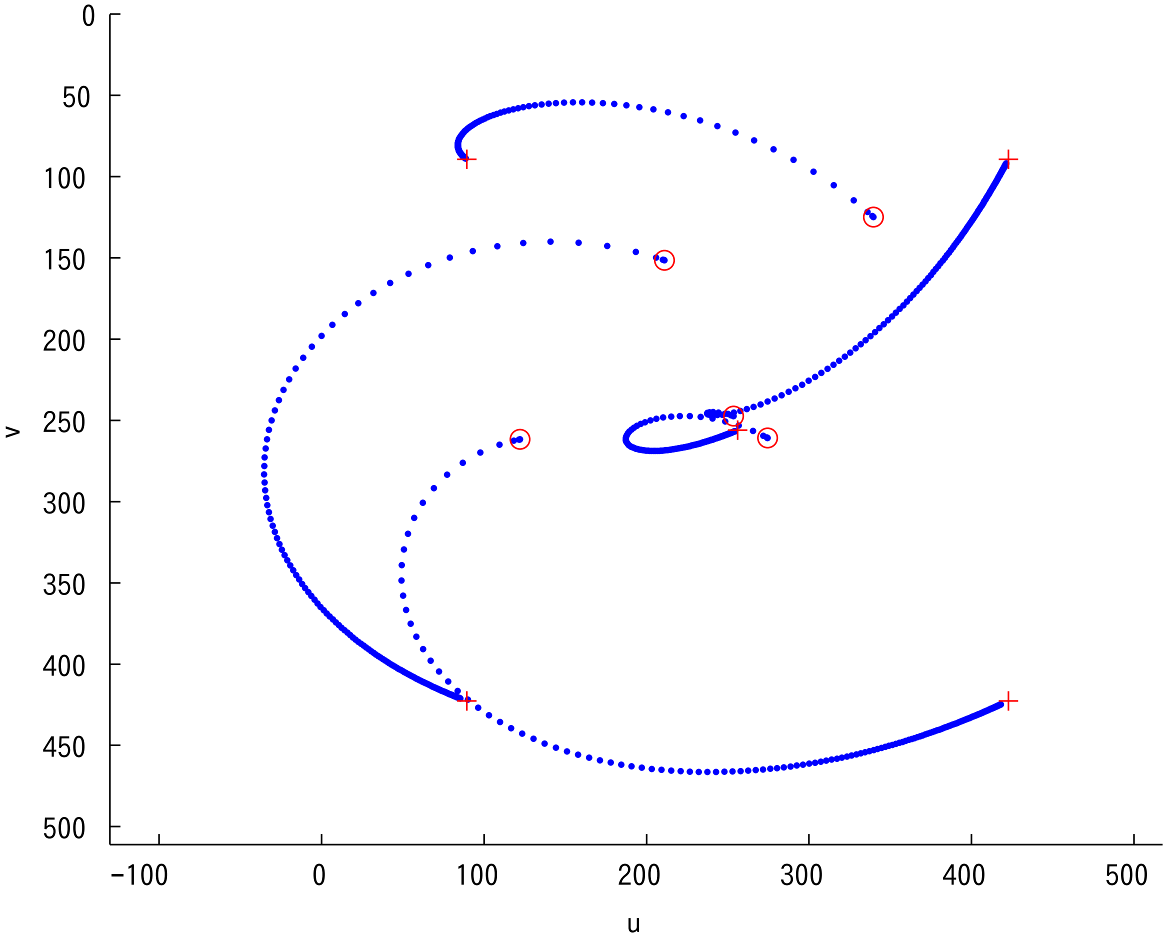
Fixed gain: Feature error



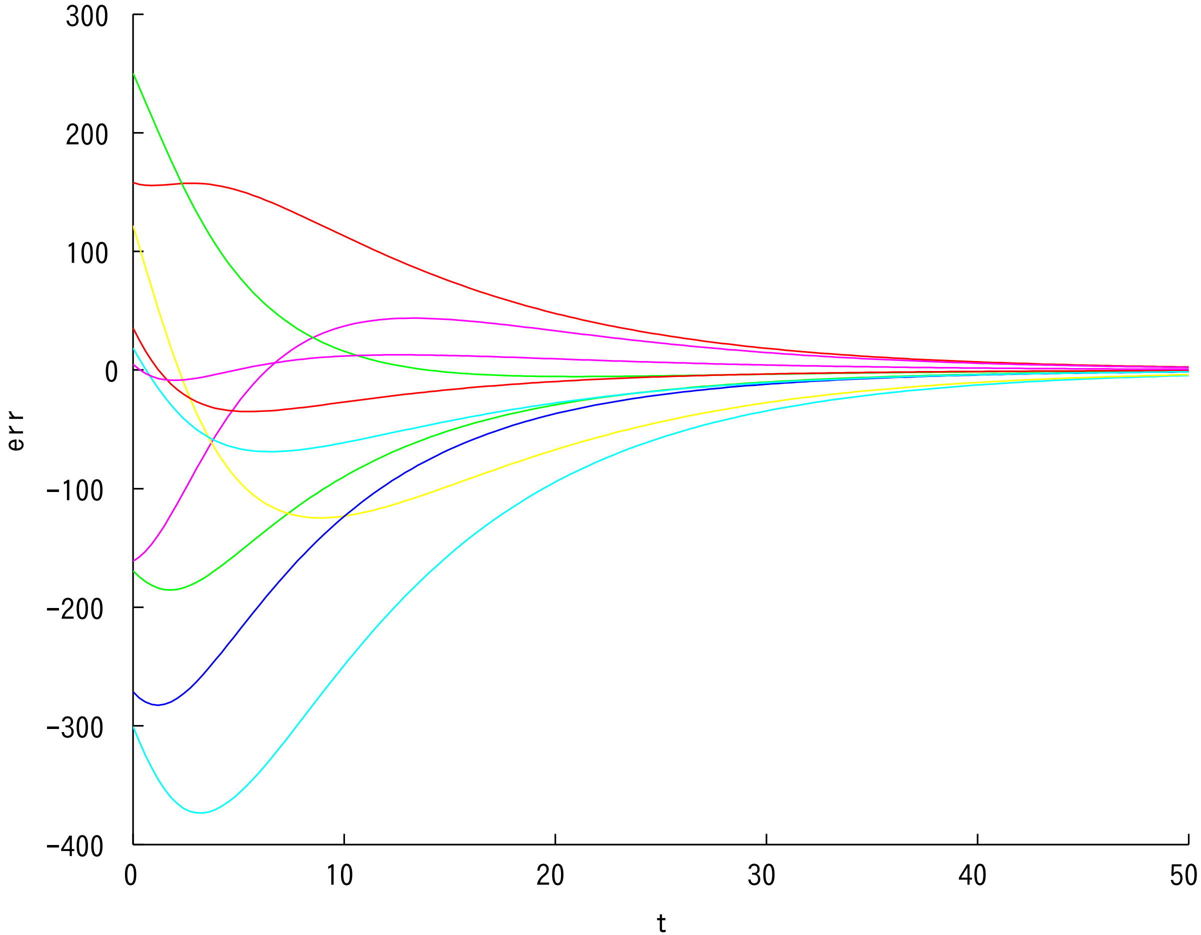
ESM: Camera trajectory



ESM: Feature trajectory



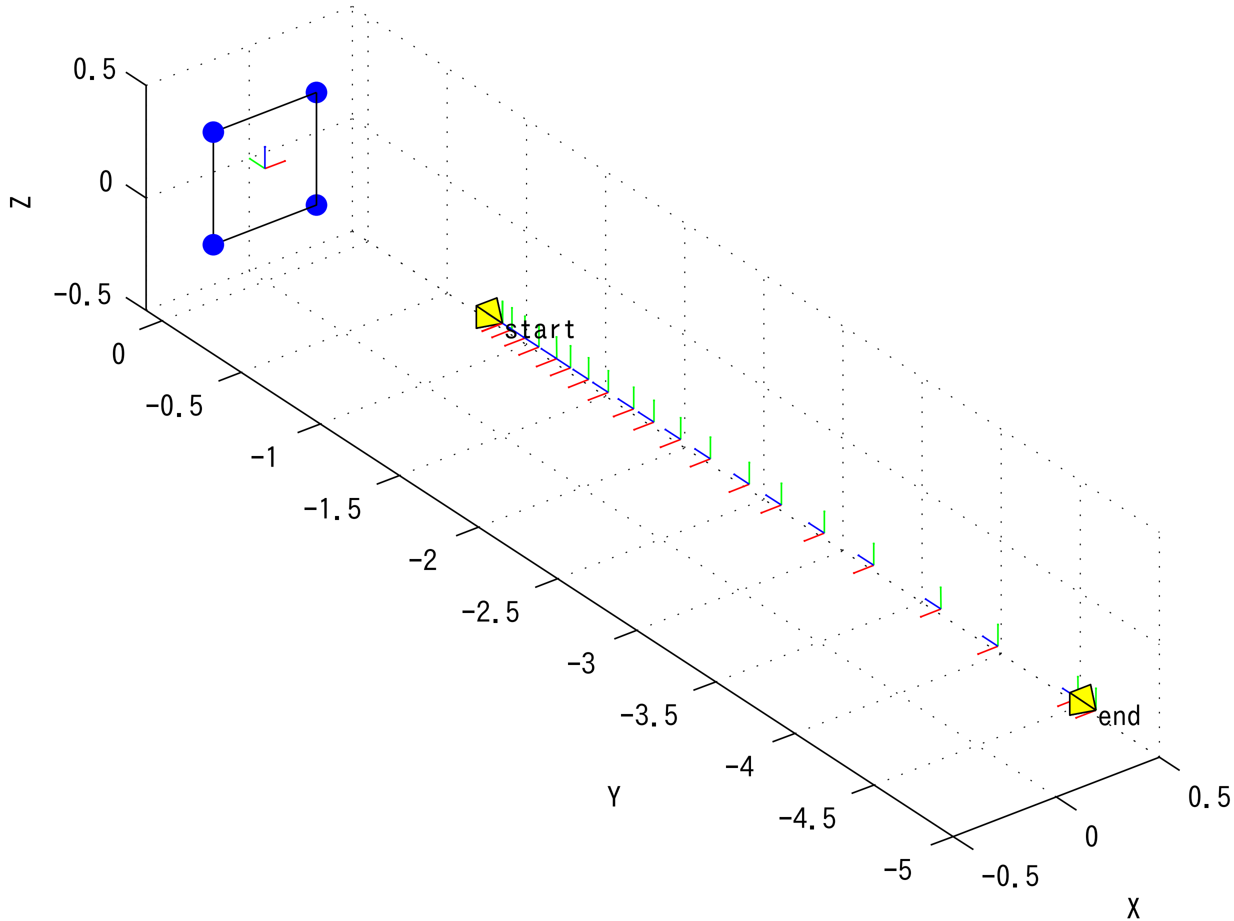
ESM: Feature error



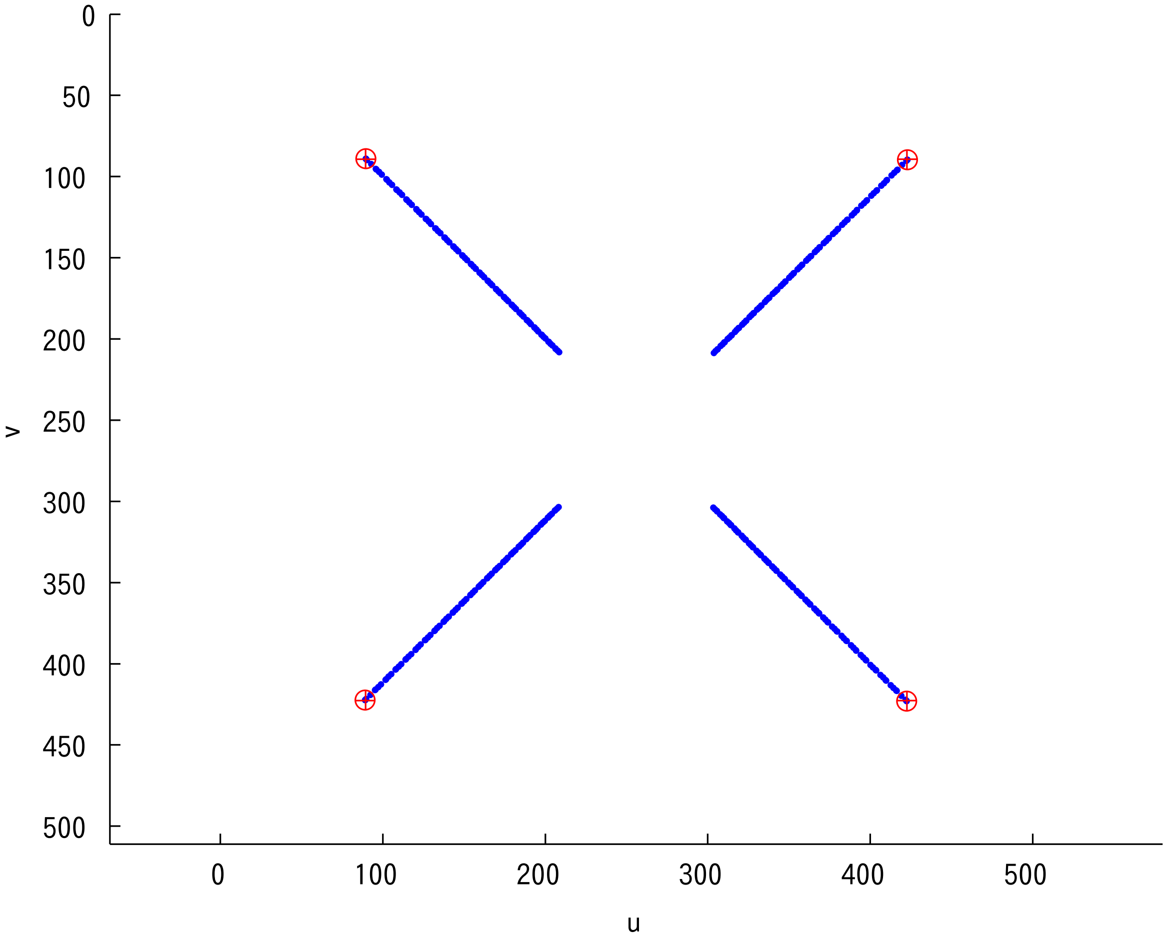
Simulation for symmetric case

```
T0c_0 =

            -1            0            0            0
             0            0            1         -1.5
             0            1            0            0
             0            0            0            1

T0c_x =

             1            0            0            0
             0            0            1         -1.5
             0           -1            0            0
             0            0            0            1
```

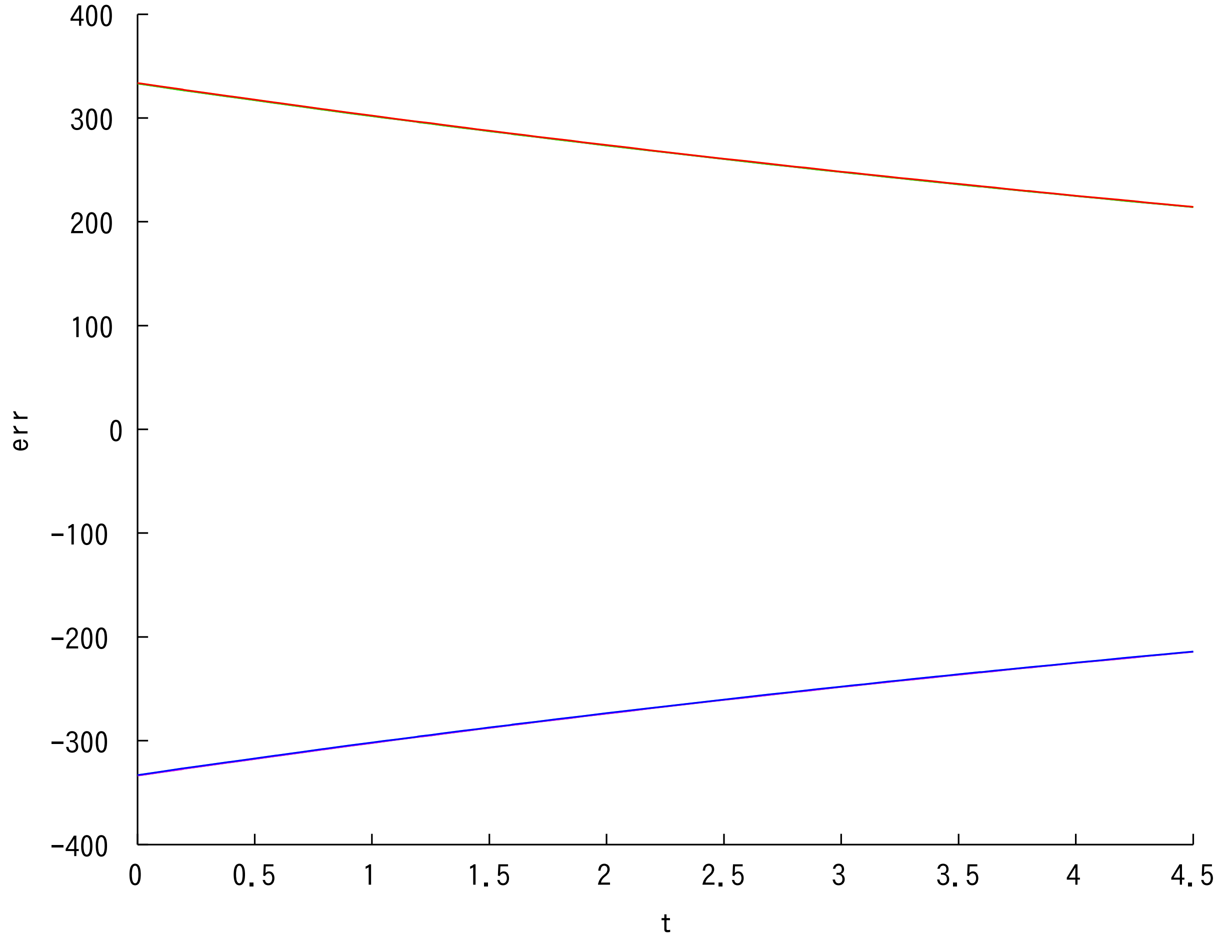
Generalized inverse: Camera trajectory



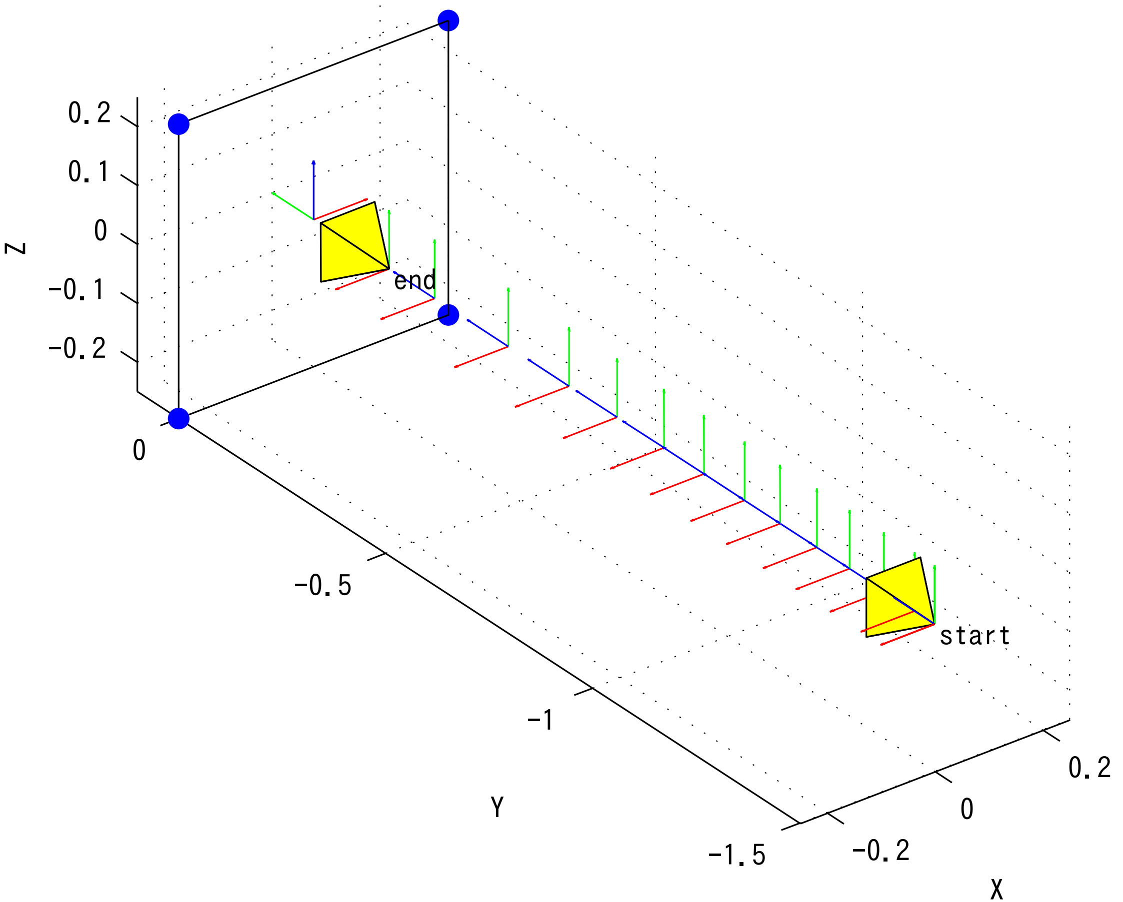
Generalized inverse: Feature trajectory



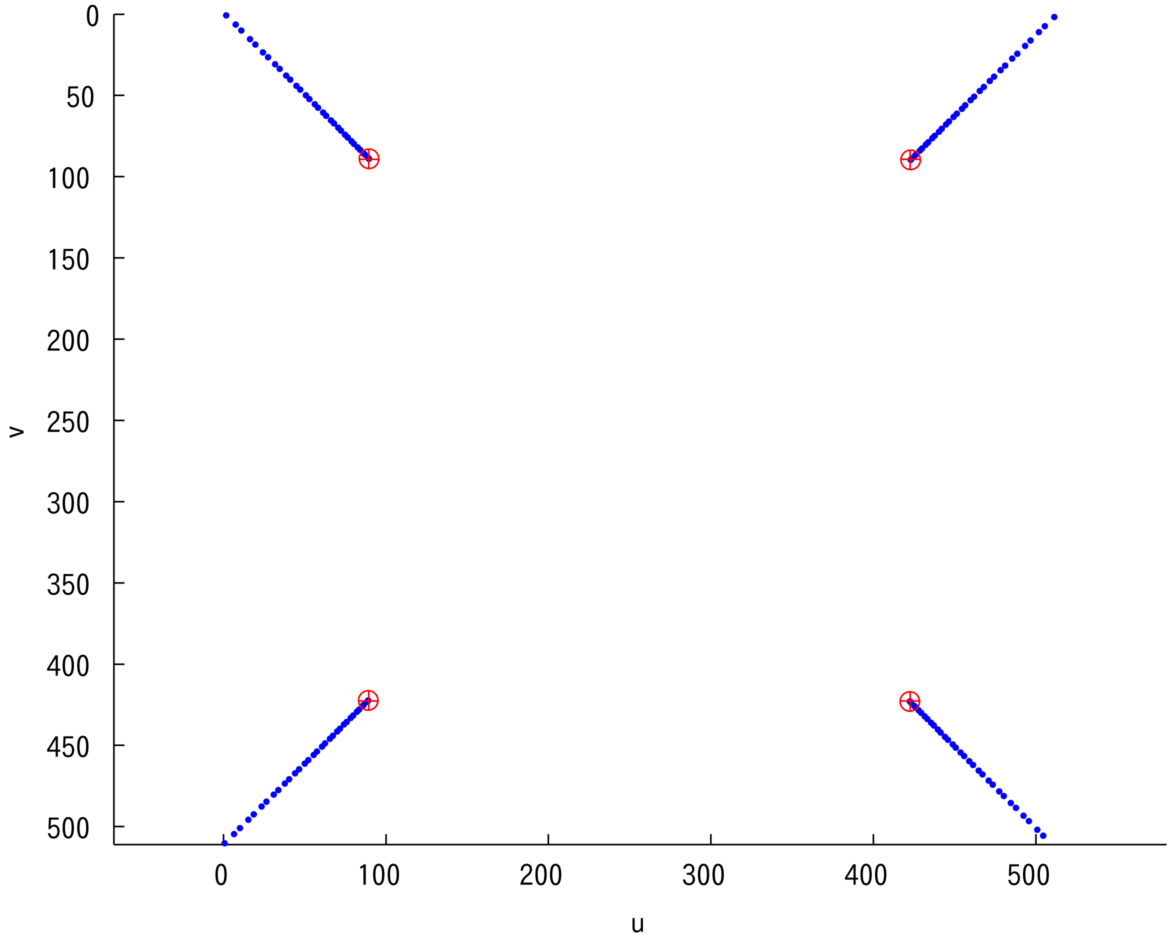
Generalized inverse: Feature error



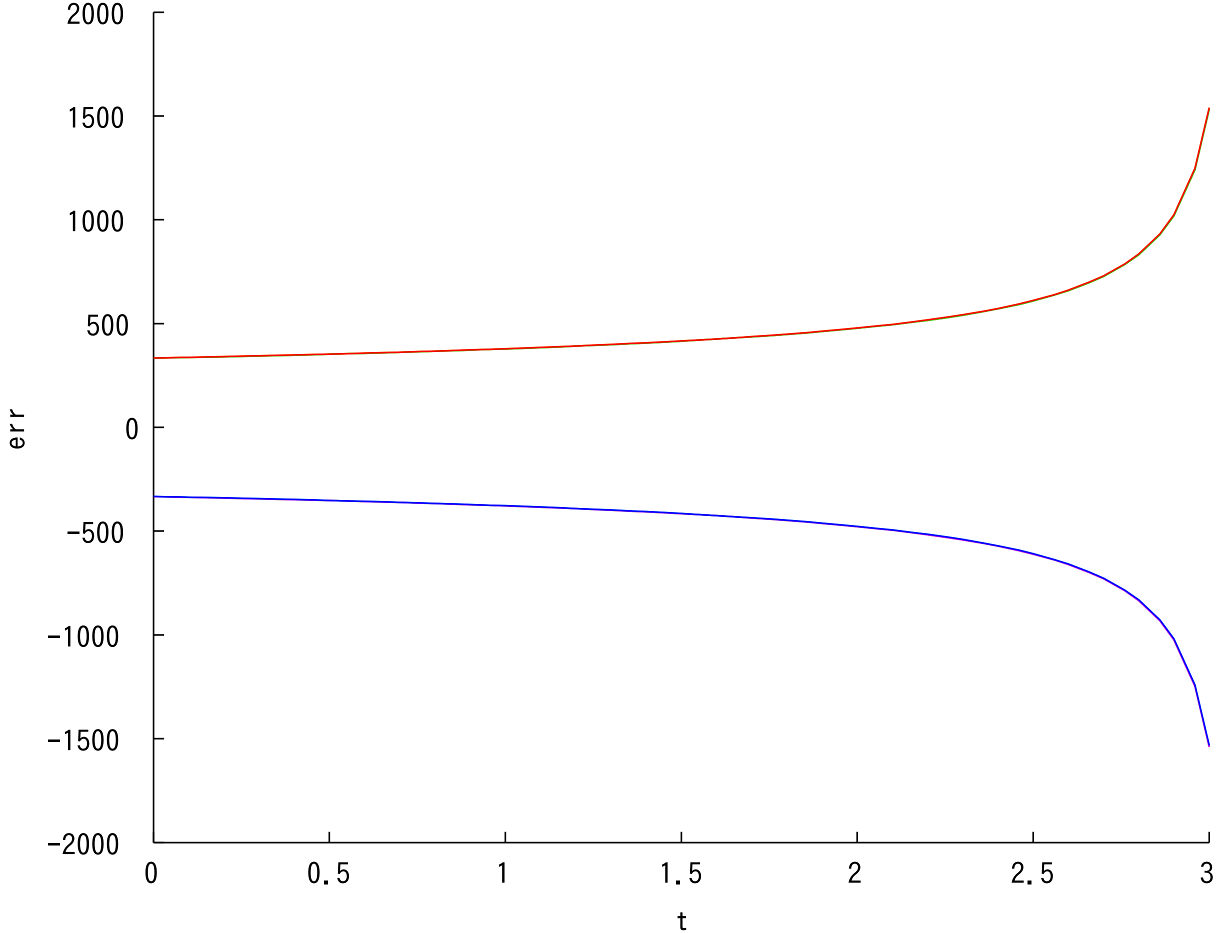
Fixed gain: Camera trajectory



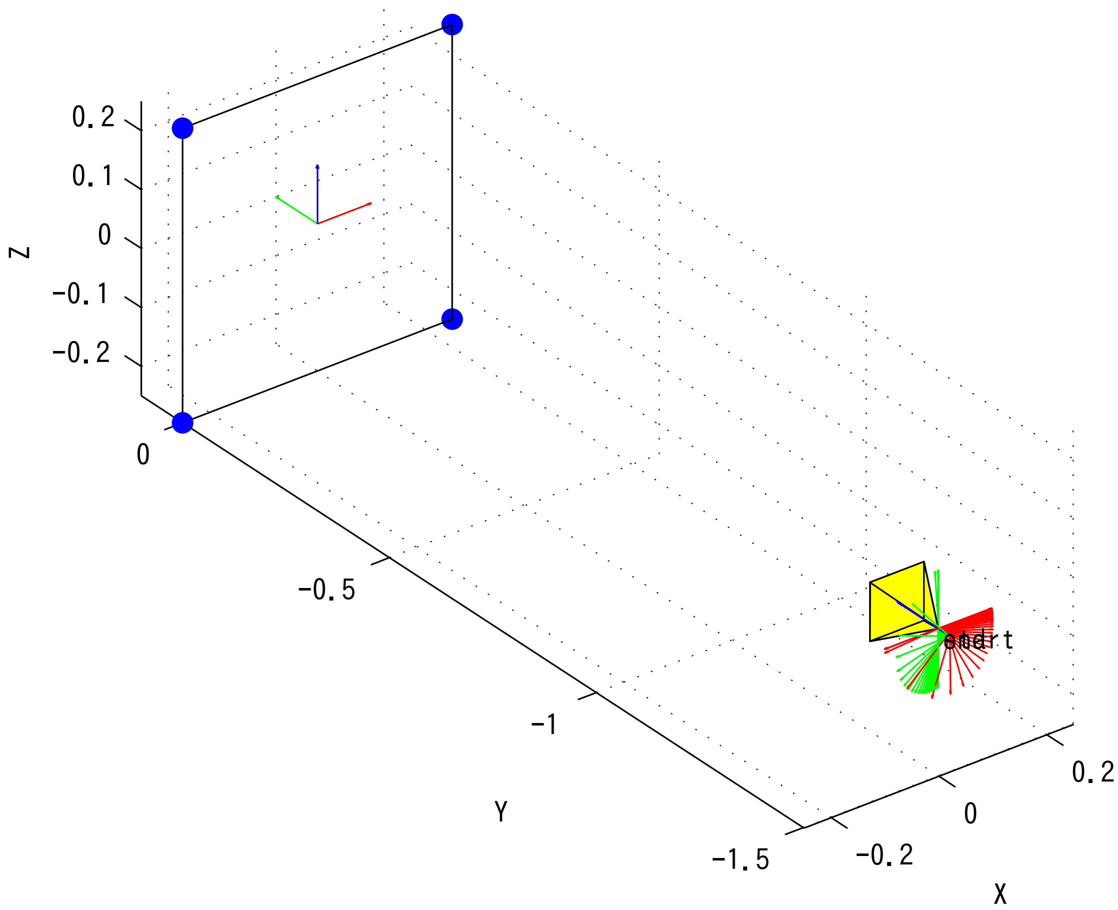
Fixed gain: Feature trajectory



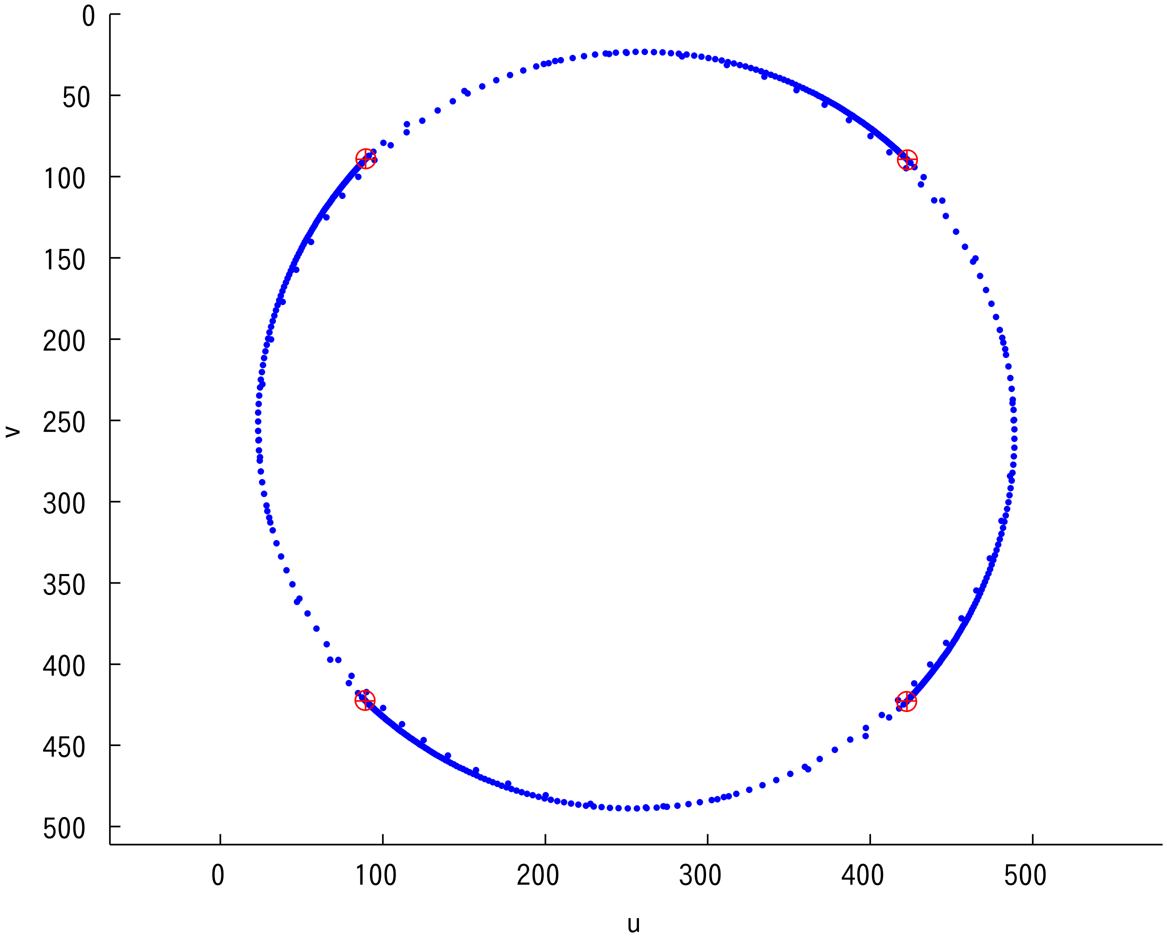
Fixed gain: Feature error



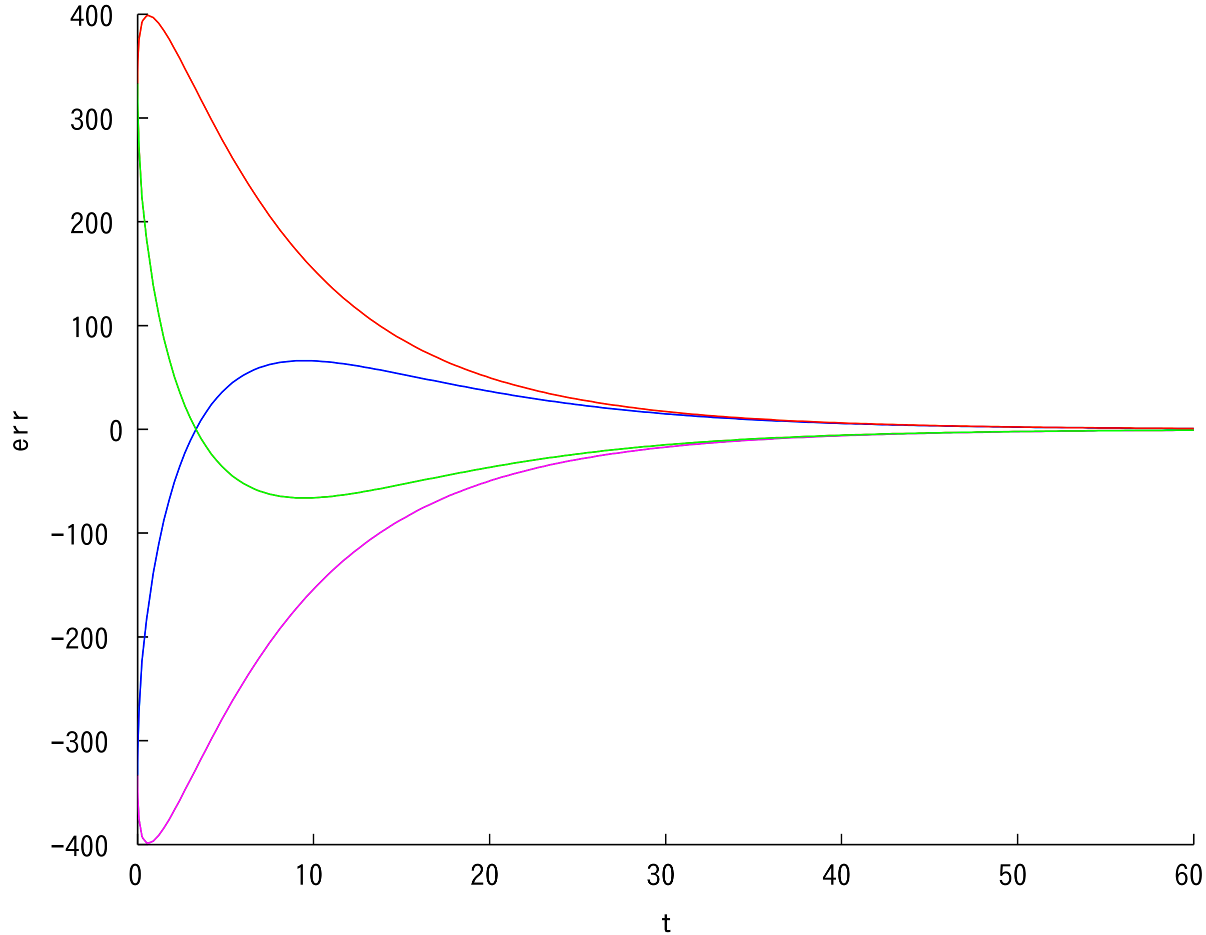
ESM: Camera trajectory



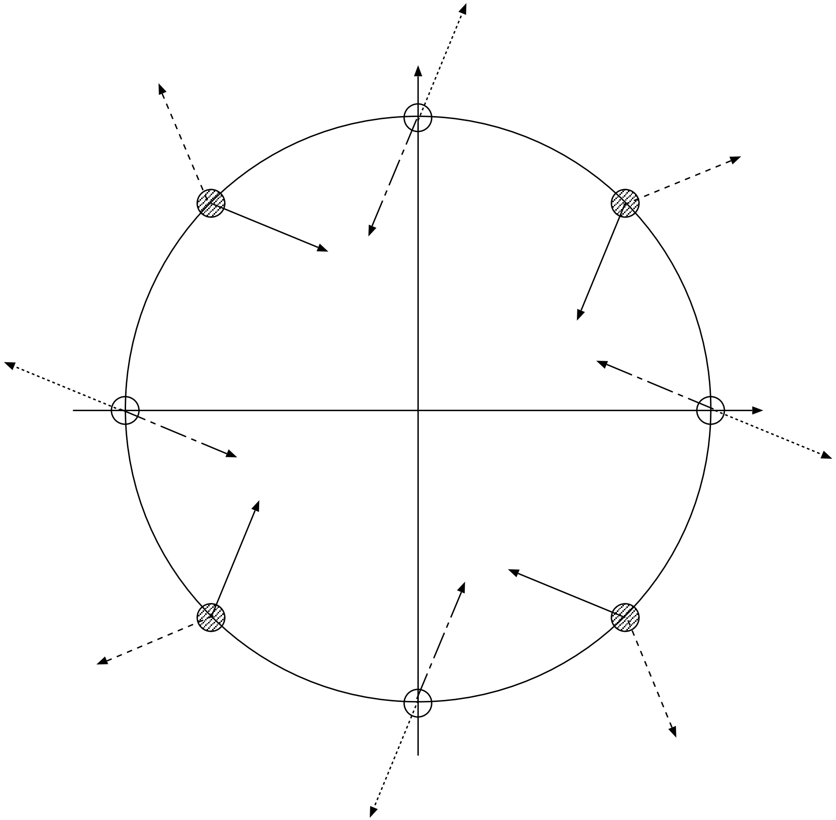
ESM: Feature trajectory



ESM: Feature error



Why fixed gain does not work properly



- 135 degree rotation
- $\mathbf{J}^{\dagger}$ law causes solid arrow flow

$$\mathbf{v} = -\lambda\ \mathbf{J}^{\dagger}(\mathbf{s} - \mathbf{s}^{*})$$

- Swap initial and desired

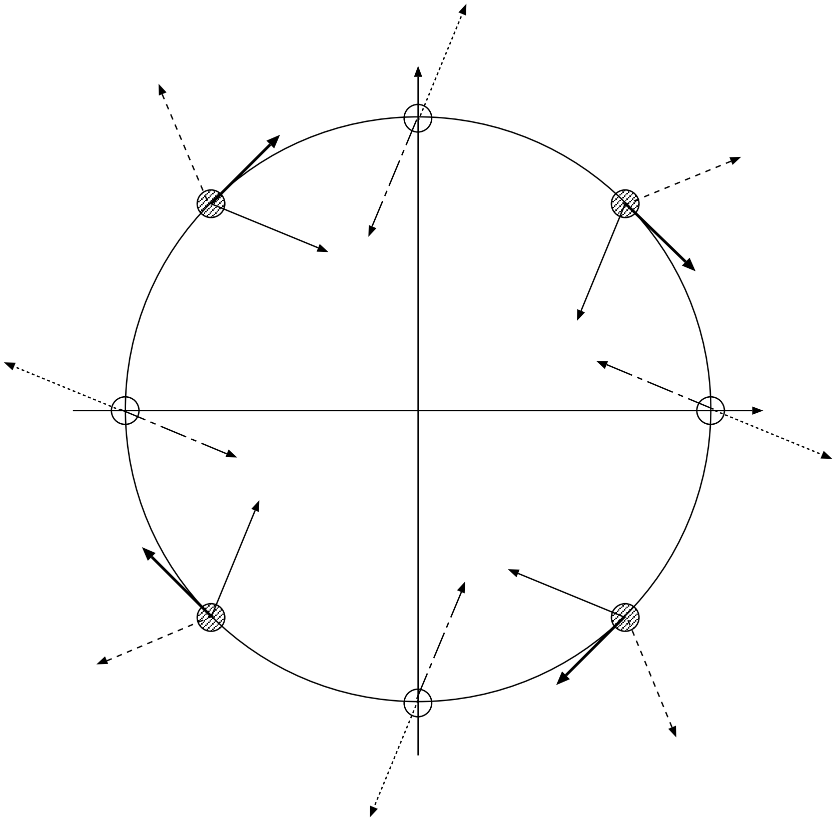
$$\mathbf{v} = -\lambda\ \mathbf{J}^{*\dagger}(\mathbf{s}^{*} - \mathbf{s})$$

then this control law generates the same solid arrow flow at $\mathbf{s}^{*}$

- The $\mathbf{J}^{*\dagger}$ law cause the inverse of solid arrow flow as indicated by the dotted arrow

$$\mathbf{v} = -\lambda\ \mathbf{J}^{*\dagger}(\mathbf{s} - \mathbf{s}^{*})$$

Why ESM work nicely



- 135 degree rotation
- ESM law

$$\mathbf{v} = -\lambda\ \mathbf{J}_{\mathsf{esm}}{}^{\dagger}(\boldsymbol{\theta})(\mathbf{s} - \mathbf{s}^*)$$

where

$$\mathbf{J}_{\mathsf{esm}}(\boldsymbol{\theta}) = (\mathbf{J}(\boldsymbol{\theta}) + \mathbf{J}^*)/2$$

- ESM is the average of $\mathbf{J}^{\dagger}$ and $\mathbf{J}^{*\dagger}$ and thus generates bold arrow flow.

Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- Sampling time issues
- ESM algorithm and visual tracking

Hybrid visual servo

- Position-based schemes have good 3D property but cannot control the image variables — easy to loose the target.
- Feature-based schemes have good robustness and good image trajectory as well as low computational cost. However the stability is local and we may have undesired motion.
- Hybrid schemes are developed to have both goodness.

Hybrid visual servo

- 2-1/2D visual servo
- Deguchi
- Corke and Hutchinson

Hybrid visual servo

- 2-1/2D visual servo
- Deguchi
- Corke and Hutchinson

2-1/2D visual servo

- Control one feature point by feature-based visual servo
- Control other DOF using position-based visual servo
- Other DOF = Depth $\rho_Z = Z/Z^*$ & Orientation $\theta\mathbf{u}$

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^*, \quad \mathbf{s} = \begin{bmatrix} x \\ y \\ \log Z \\ \theta\mathbf{u} \end{bmatrix}, \quad \mathbf{s}^* = \begin{bmatrix} x^* \\ y^* \\ \log Z^* \\ \mathbf{0} \end{bmatrix}$$

- The third element is $e_z = \log \rho_Z$
- Note that $\theta\mathbf{u}$ is obtained by Homography and

$$\rho_Z = \det(\mathbf{H}) \frac{\mathbf{n}^{*\top}\mathbf{m}^*}{\mathbf{n}^\top\mathbf{m}}$$

2-1/2D visual servo

- Controlled feature point: $\mathbf{x} = [x \ y]^\top$
- Derivative relations

$$
\begin{aligned}
\dot{\mathbf{x}} &= \mathbf{J}_{\mathbf{x}} \mathbf{v} \\
\mathbf{J}_{\mathbf{x}} &= \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix} \\
\mathbf{v} &= \begin{bmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{bmatrix}
\end{aligned}
$$

$$
\frac{d}{dt} \log Z = \frac{\dot{Z}}{Z} = \frac{1}{Z} \begin{bmatrix} 0 & 0 & -1 & -y & x & 0 \end{bmatrix} \mathbf{v}
$$

$$
\frac{d(\theta \mathbf{u})}{dt} = \mathbf{J}_{\theta \mathbf{u}} \boldsymbol{\omega}_c, \quad \mathbf{J}_{\theta \mathbf{u}} = \mathbf{I} - \frac{\theta}{2} [\mathbf{u}]_\wedge + \left(1 - \frac{\mathrm{sinc}\theta}{\mathrm{sinc}^2 \frac{\theta}{2}} \right) [\mathbf{u}]_\wedge^2
$$

2-1/2D visual servo

- In summary, we have

$$\dot{\mathbf{e}} = \mathbf{J}\mathbf{v} = \begin{bmatrix} \mathbf{J}_v & \mathbf{J}_\omega \\ \mathbf{0} & \mathbf{J}_{\theta\mathbf{u}} \end{bmatrix} \begin{bmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{bmatrix}$$

where

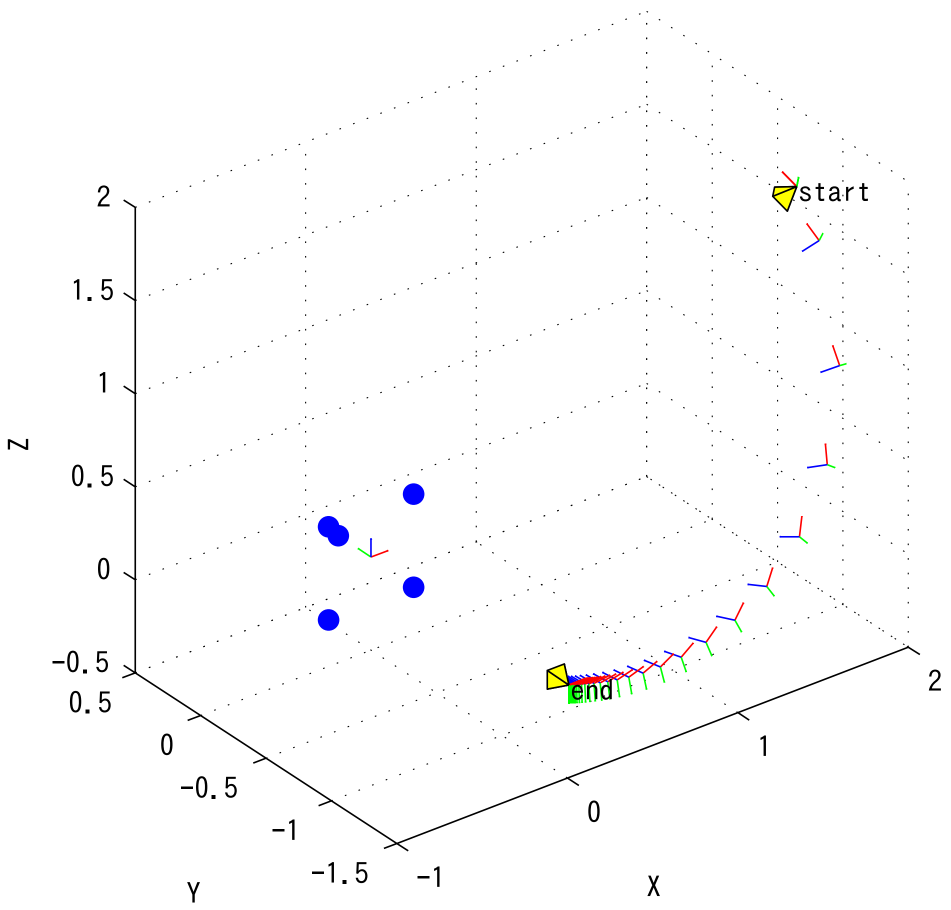
$$\begin{aligned} \mathbf{J}_v &= \frac{1}{Z^*\rho_Z} \begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1 \end{bmatrix}, \\ \mathbf{J}_\omega &= \begin{bmatrix} xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0 \end{bmatrix} \end{aligned}$$

- Control law

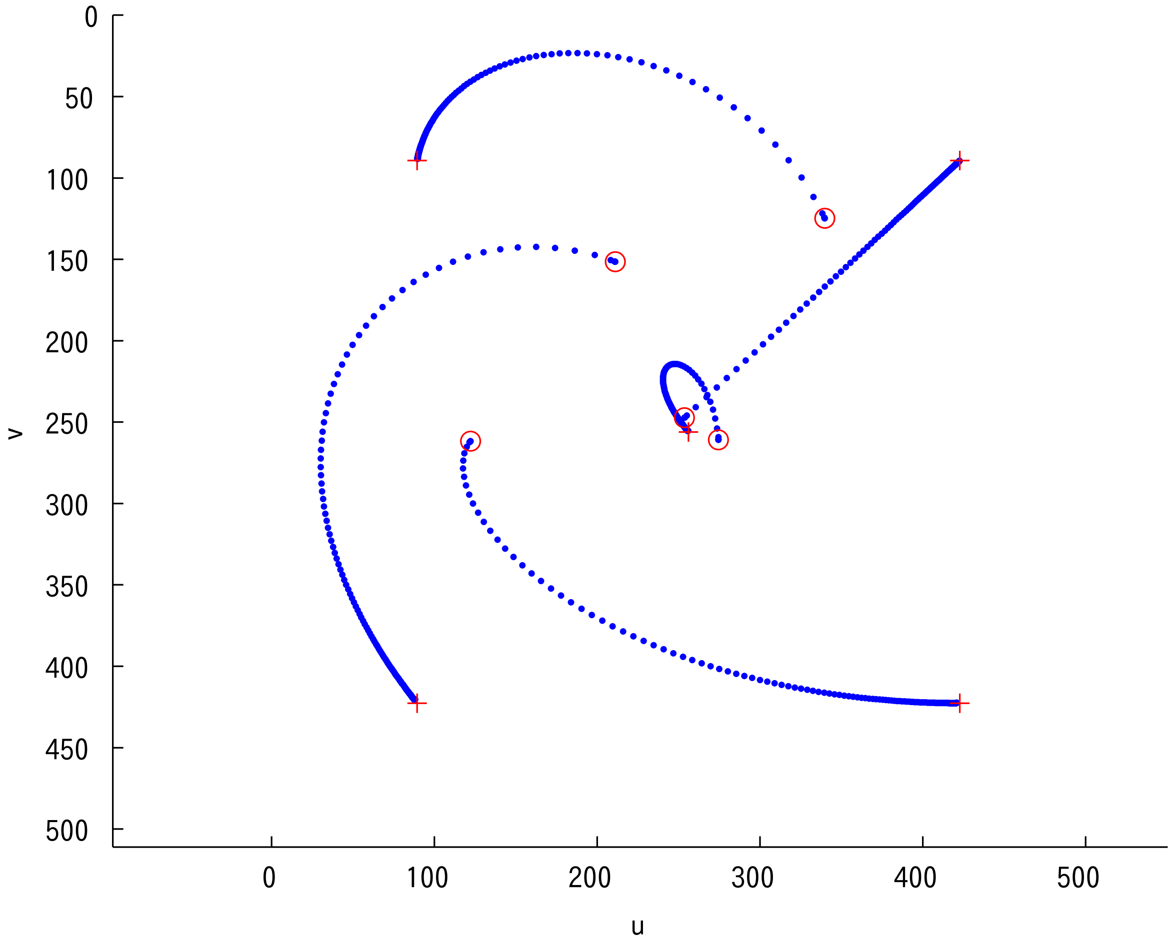
$$\mathbf{v} = \mathbf{J}^{-1}\mathbf{e}$$

```
T0c_0 =

   -0.46742      0.67153     -0.57495       1.6598
    0.49873      0.73729       0.4557        -1.09
    0.72992    -0.073743     -0.67954       1.9059
          0            0            0            1

T0c_x =

          1            0            0            0
          0            0            1         -1.5
          0           -1            0            0
          0            0            0            1
```

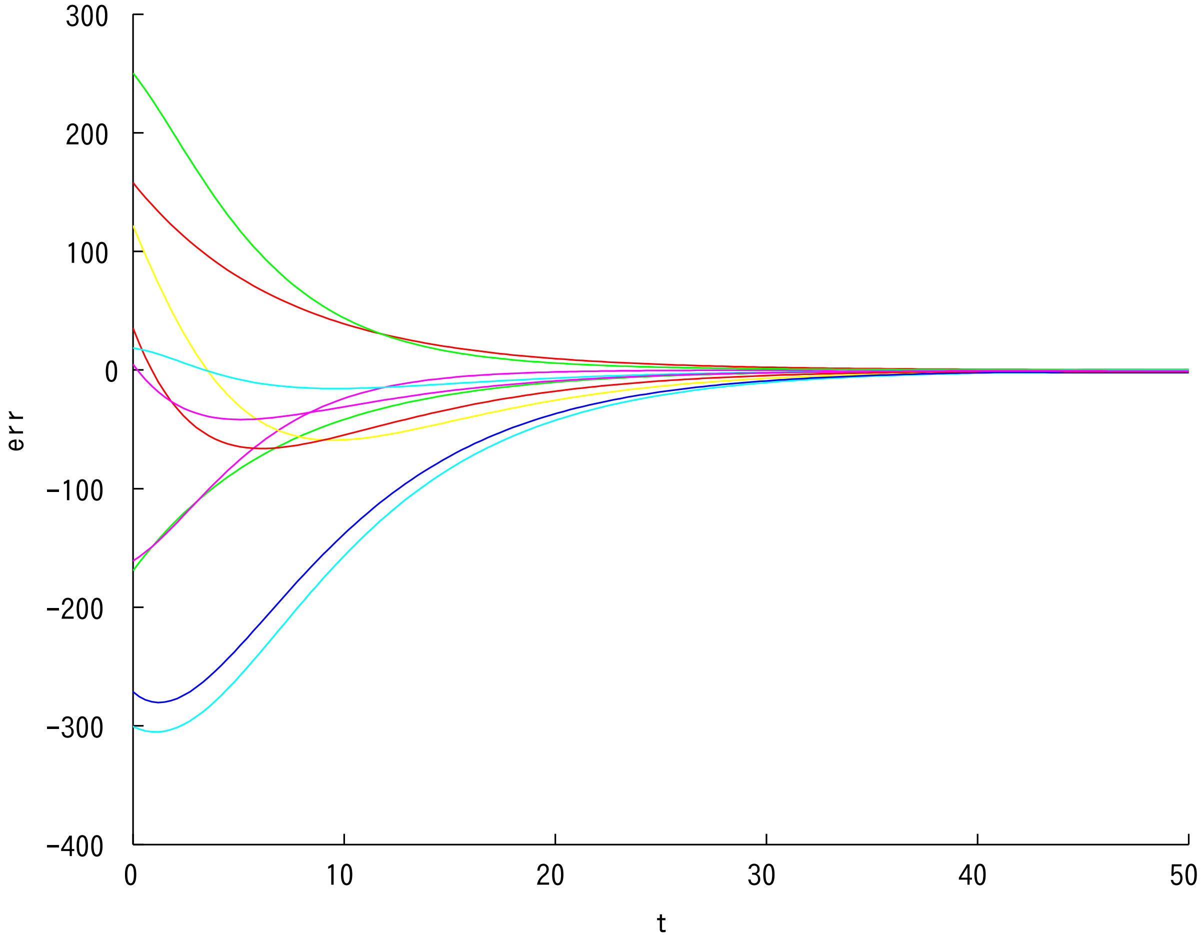
2-1/2D hybrid: Camera trajectory



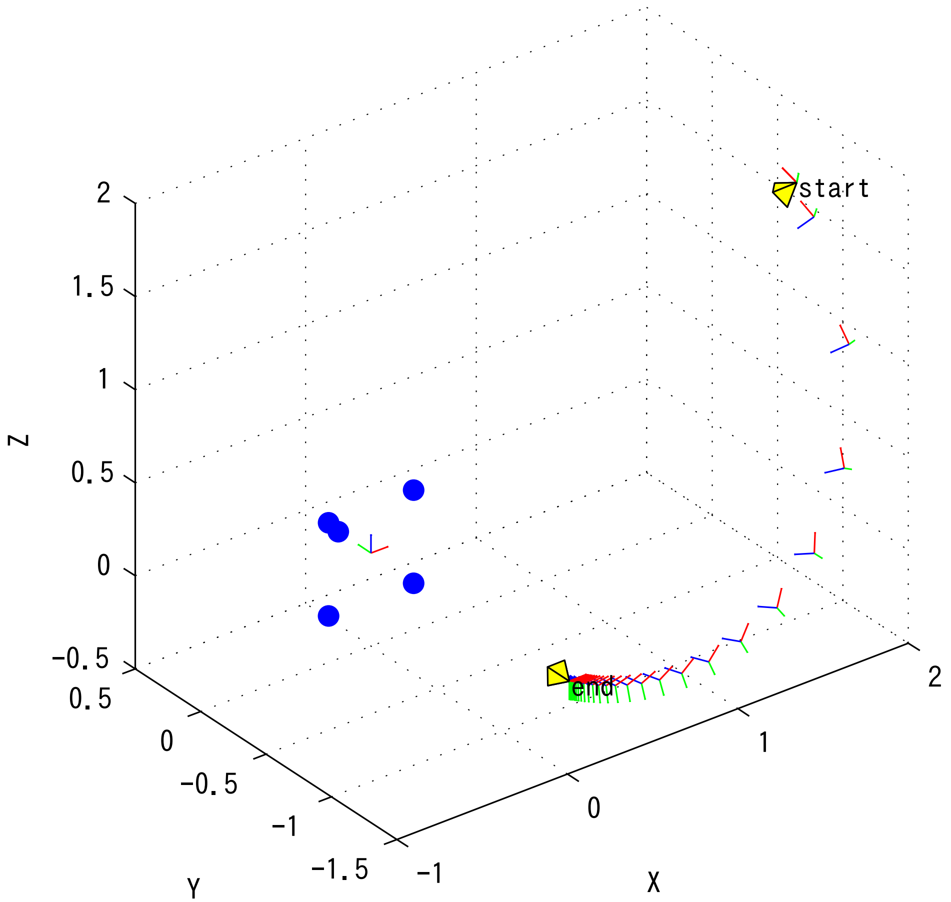
2-1/2D hybrid: Feature trajectory



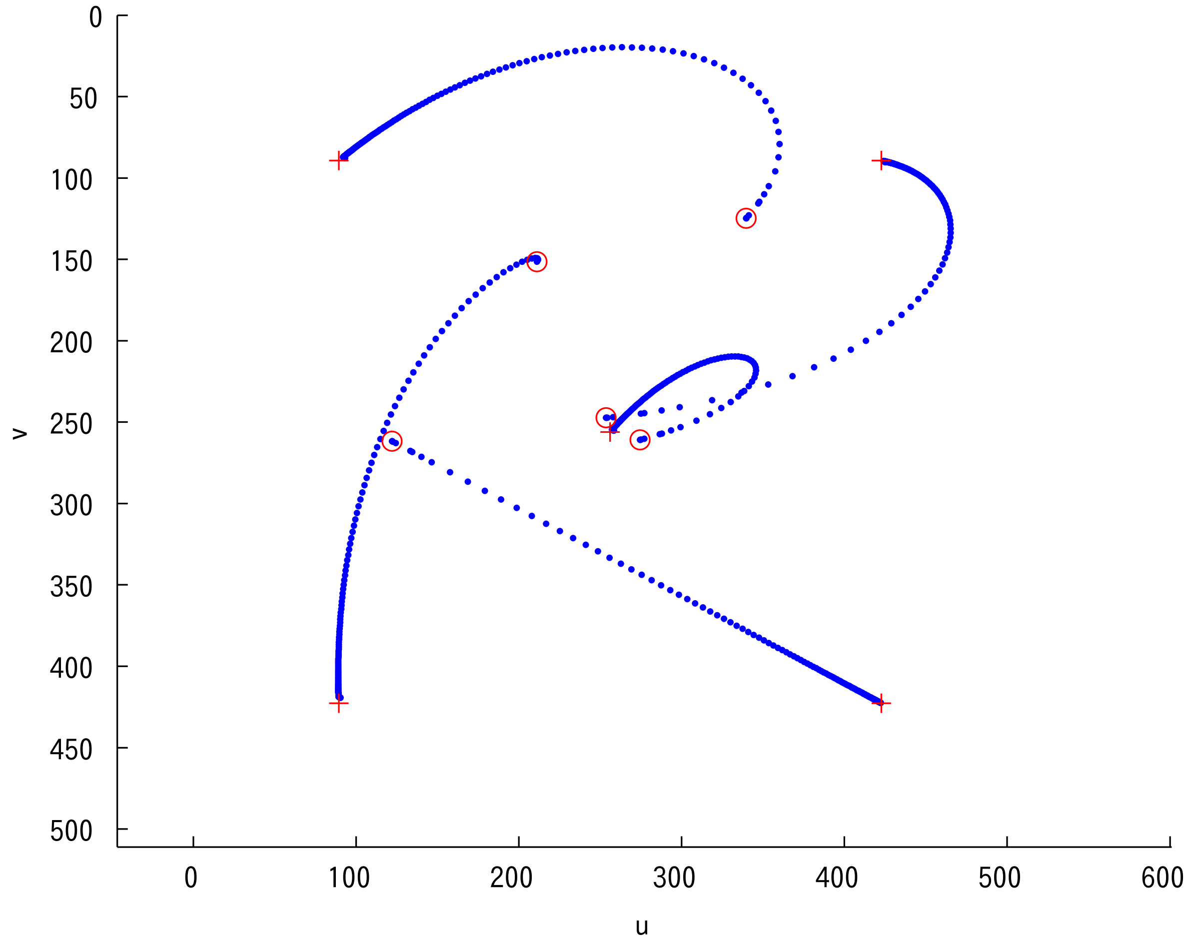
2-1/2D hybrid: Feature error



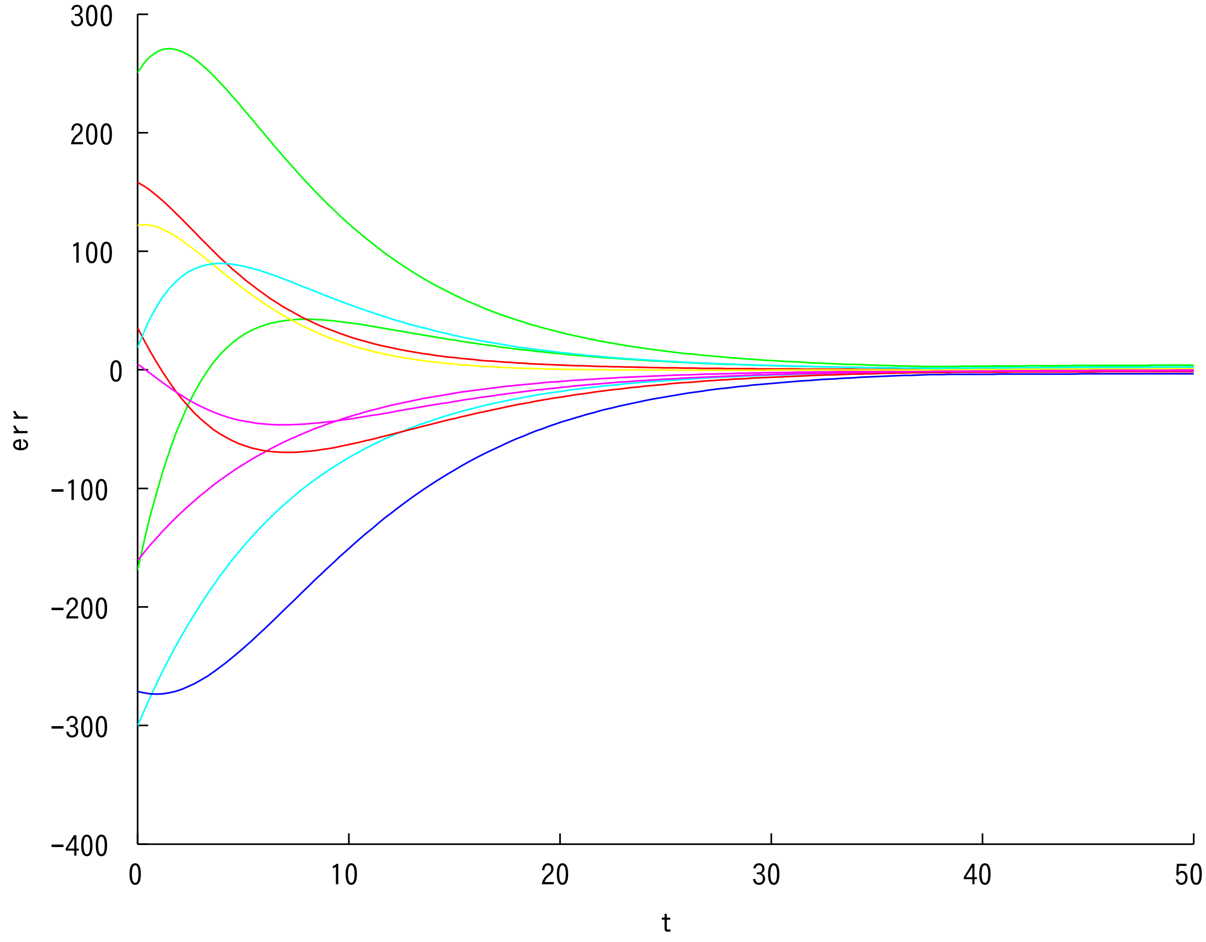
2-1/2D hybrid: Camera trajectory



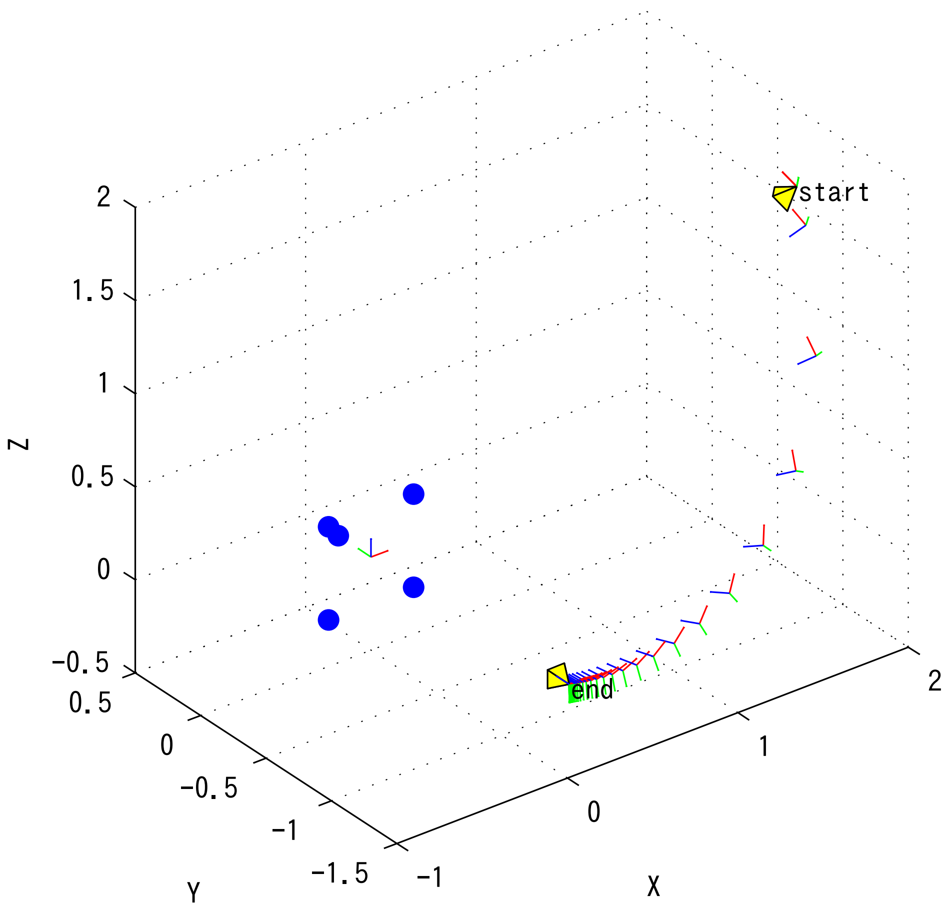
2-1/2D hybrid: Feature trajectory



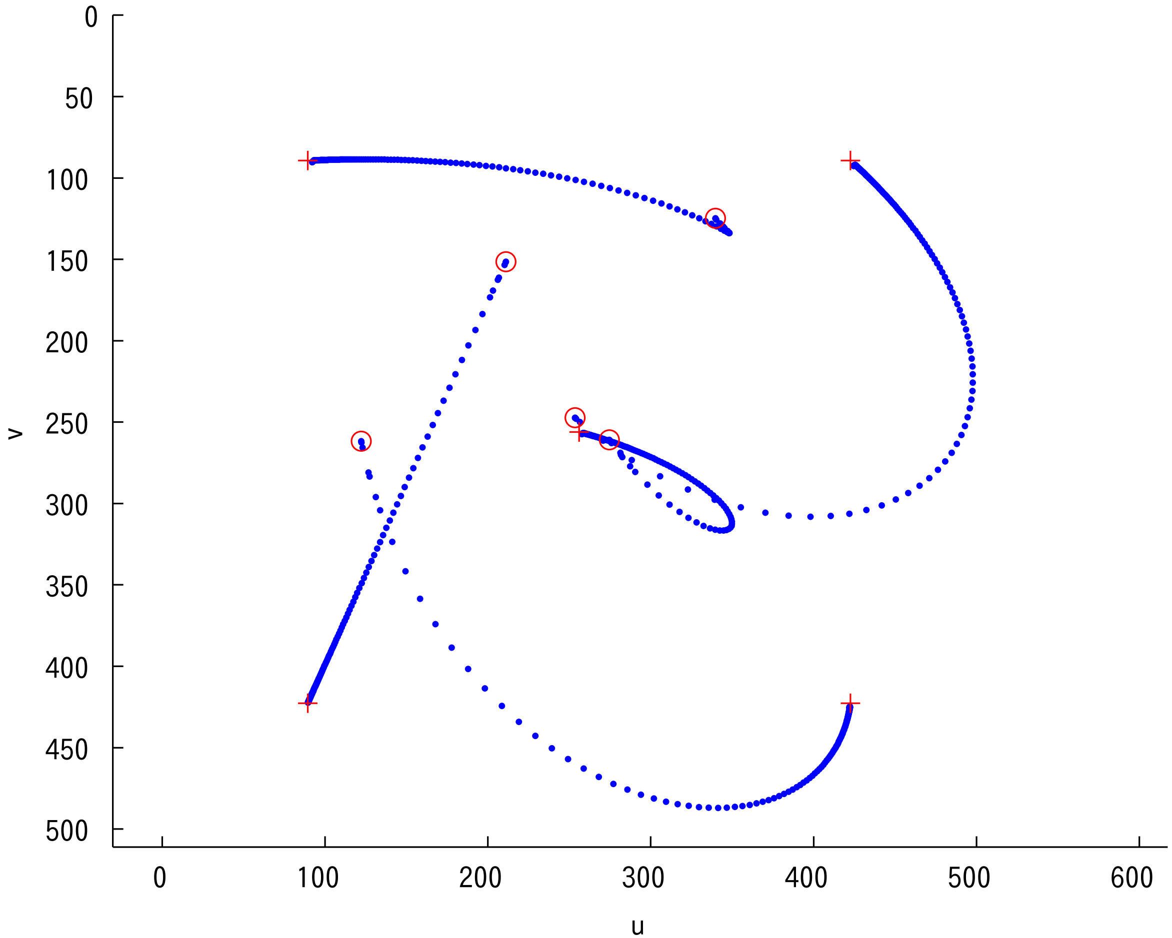
2-1/2D hybrid: Feature error



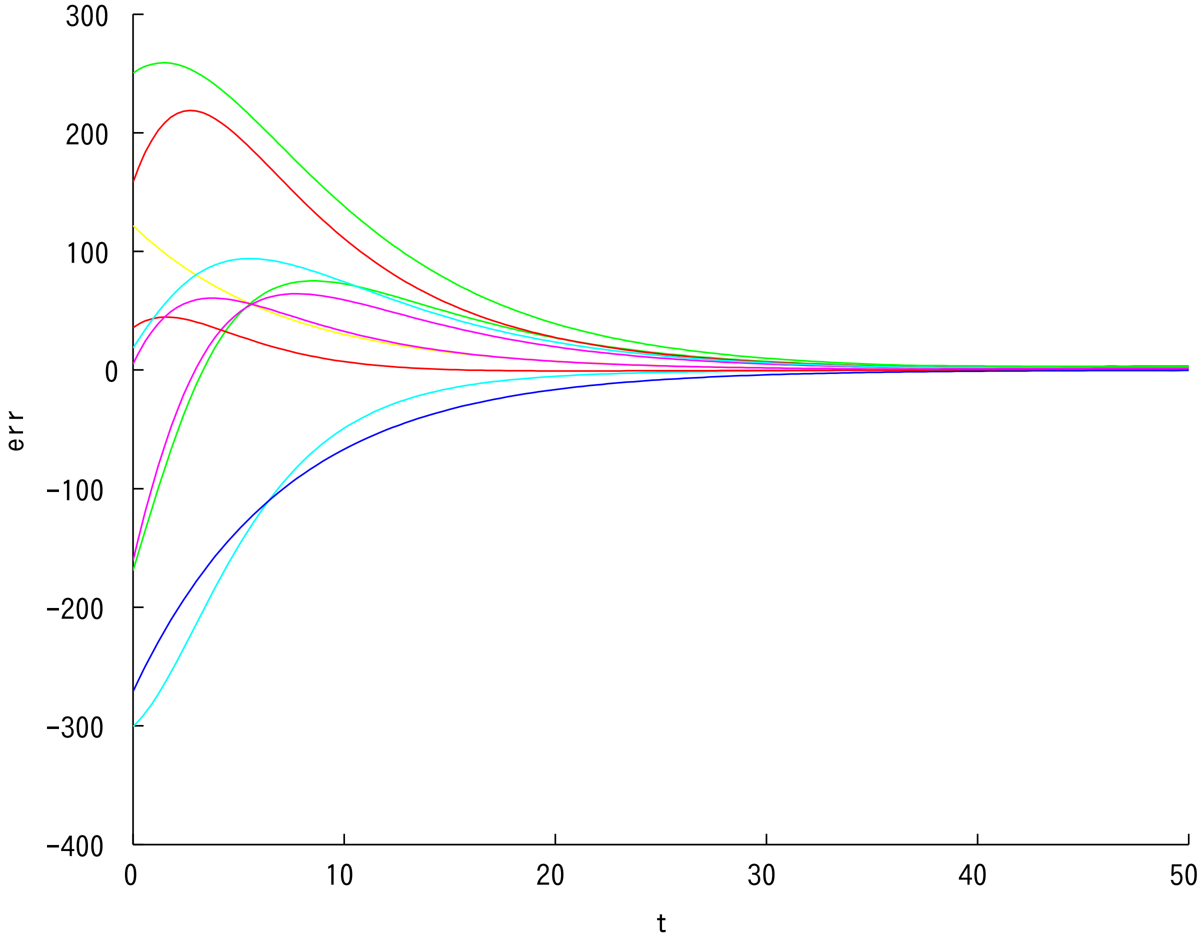
2-1/2D hybrid: Camera trajectory



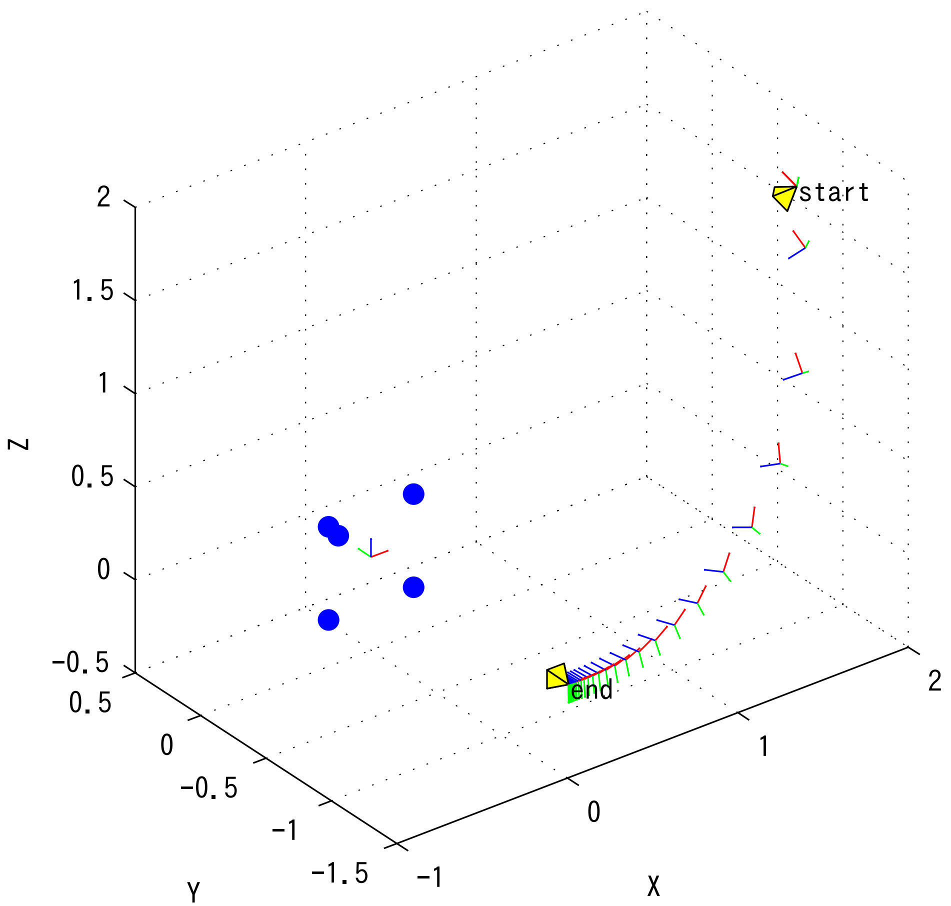
2-1/2D hybrid: Feature trajectory



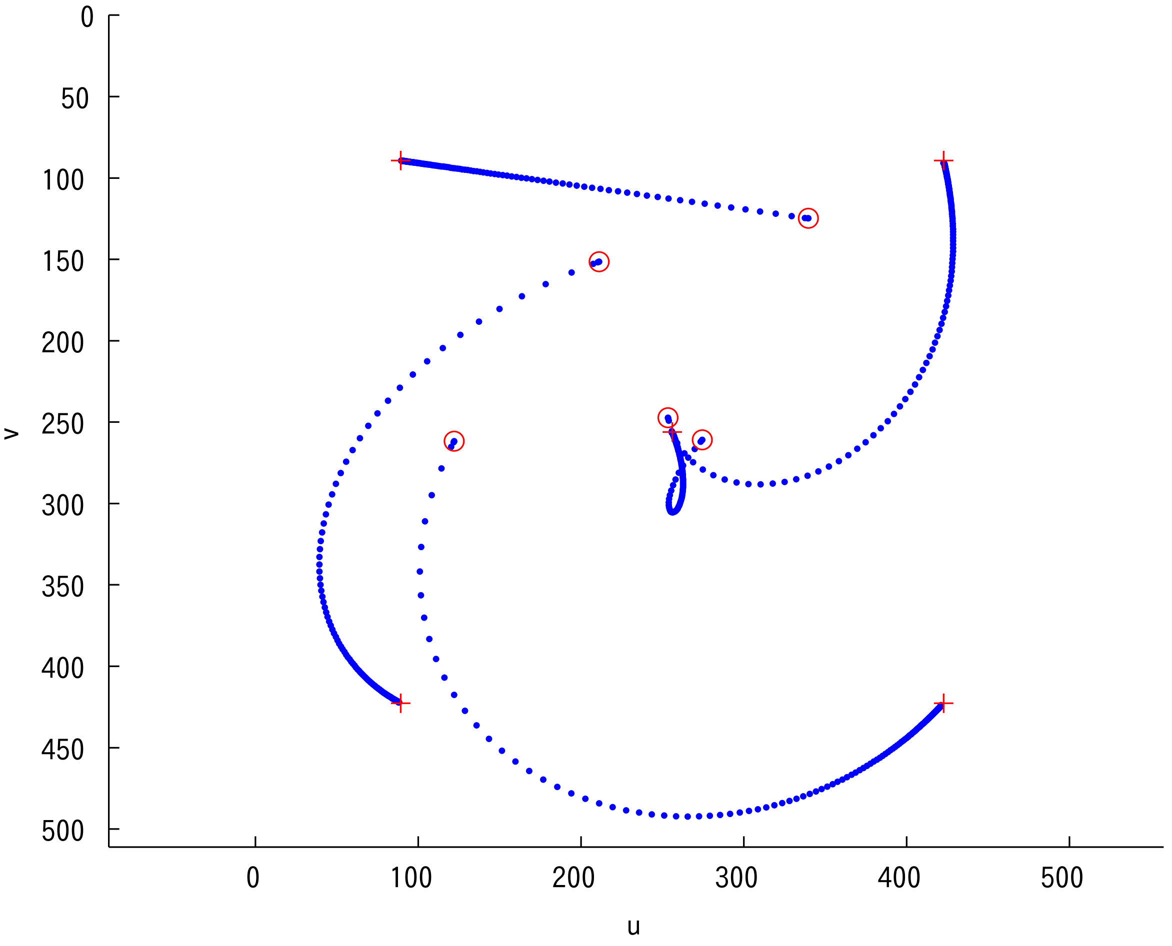
2-1/2D hybrid: Feature error



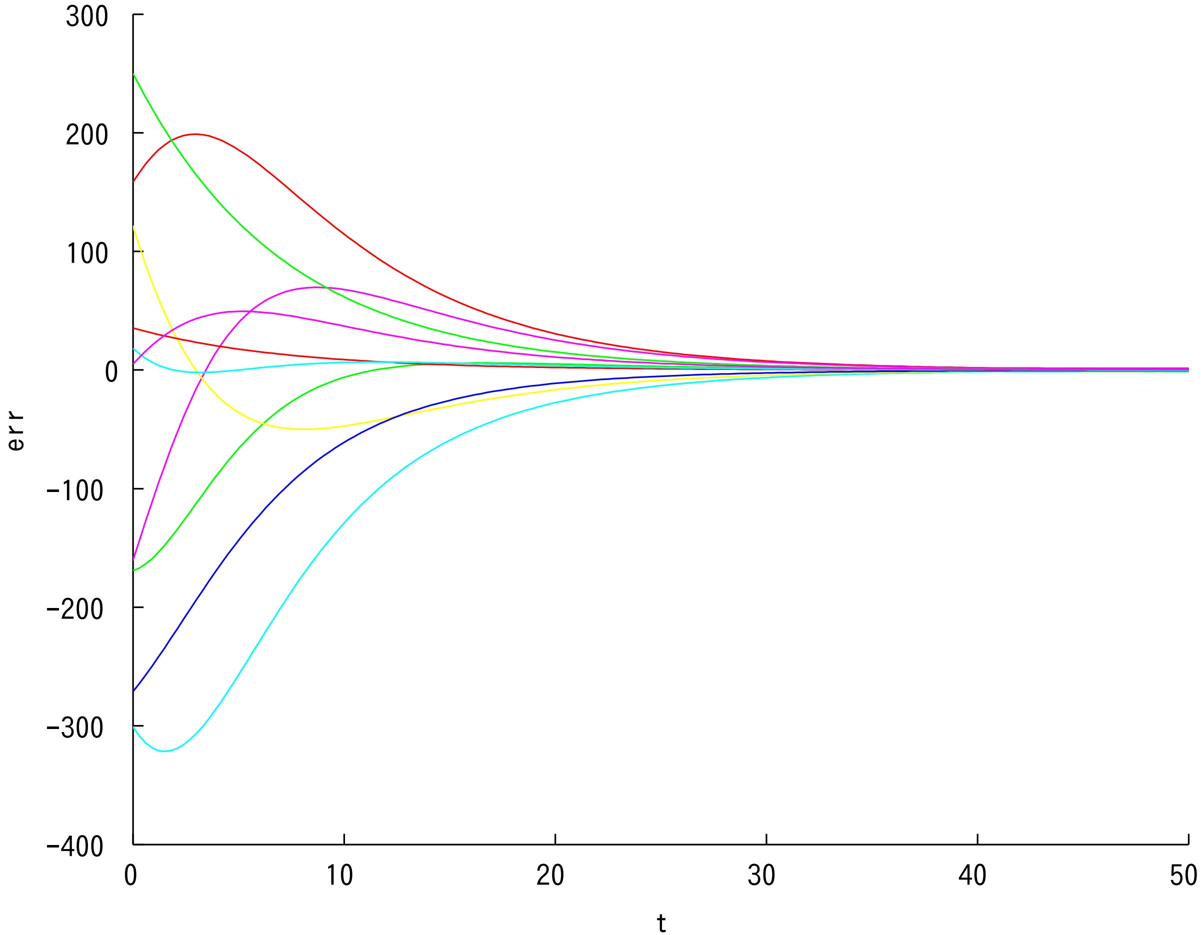
2-1/2D hybrid: Camera trajectory



2-1/2D hybrid: Feature trajectory



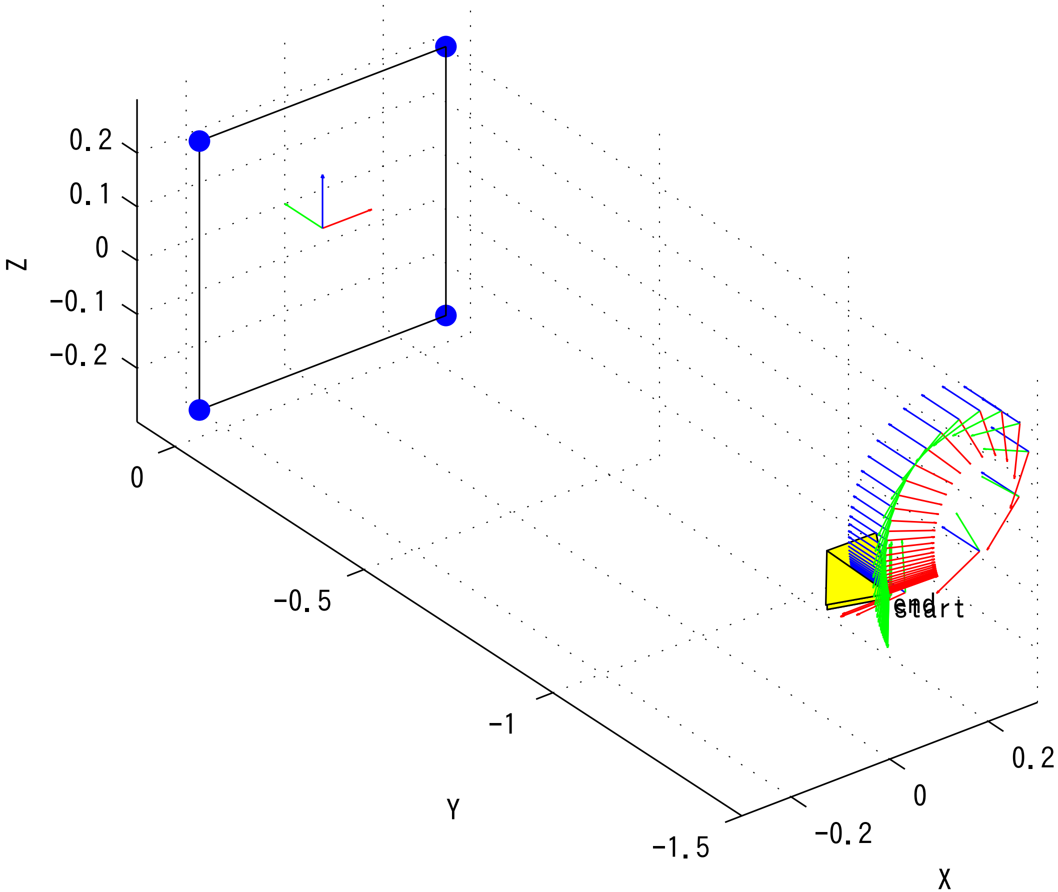
2-1/2D hybrid: Feature error



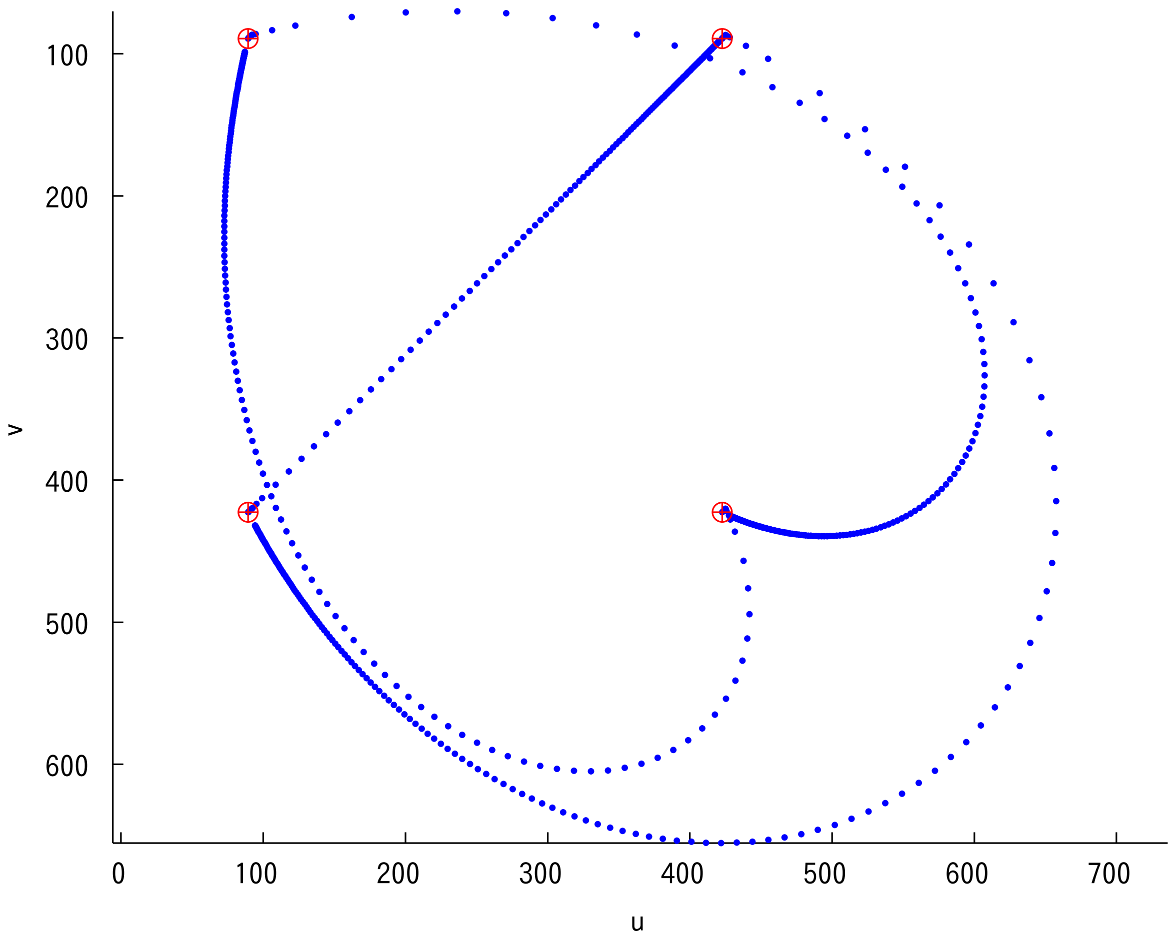
Simulation for symmetric case

```
T0c_0 =

          -1           0           0           0
           0           0           1        -1.5
           0           1           0           0
           0           0           0           1

T0c_x =

           1           0           0           0
           0           0           1        -1.5
           0          -1           0           0
           0           0           0           1
```

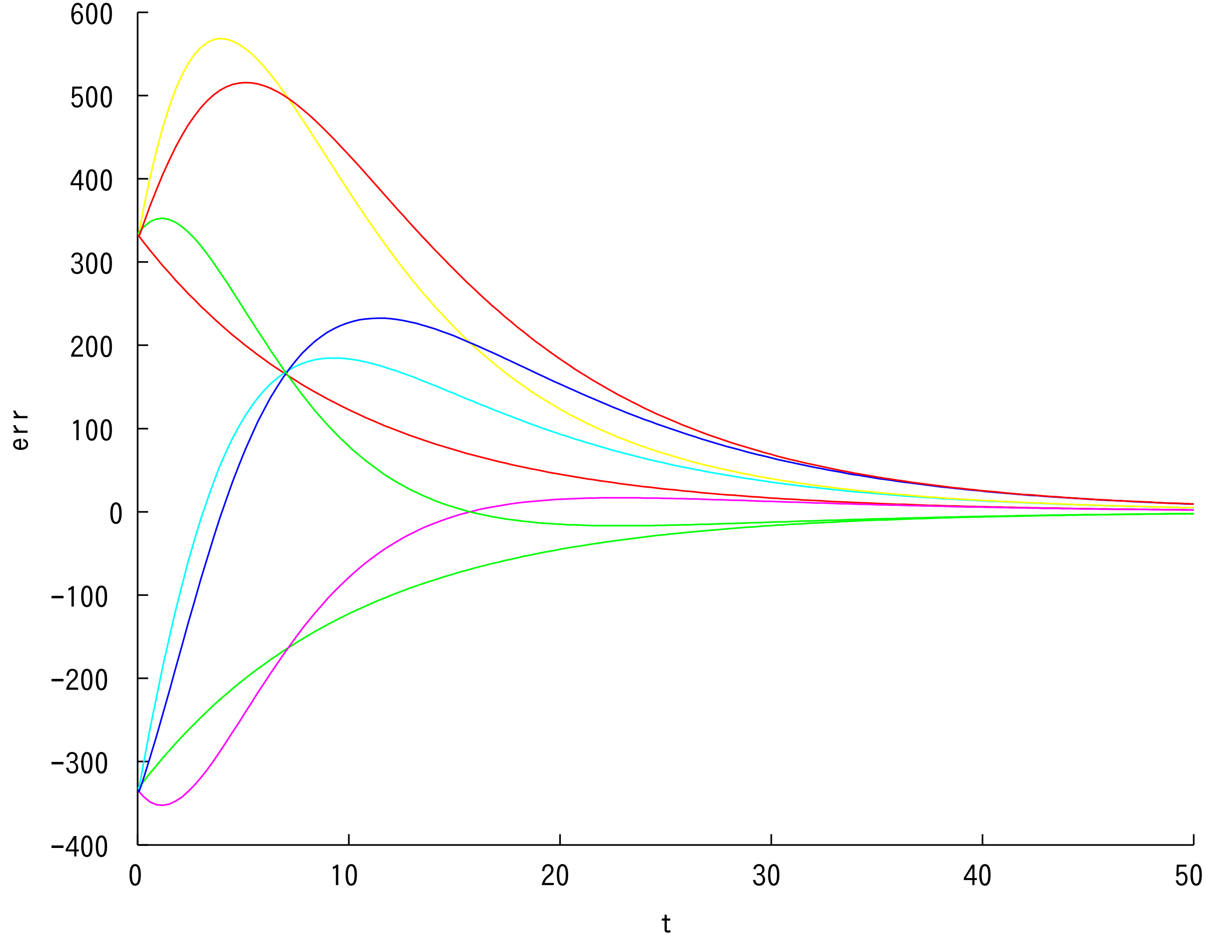
2-1/2D hybrid: Camera trajectory



2-1/2D hybrid: Feature trajectory



2-1/2D hybrid: Feature error

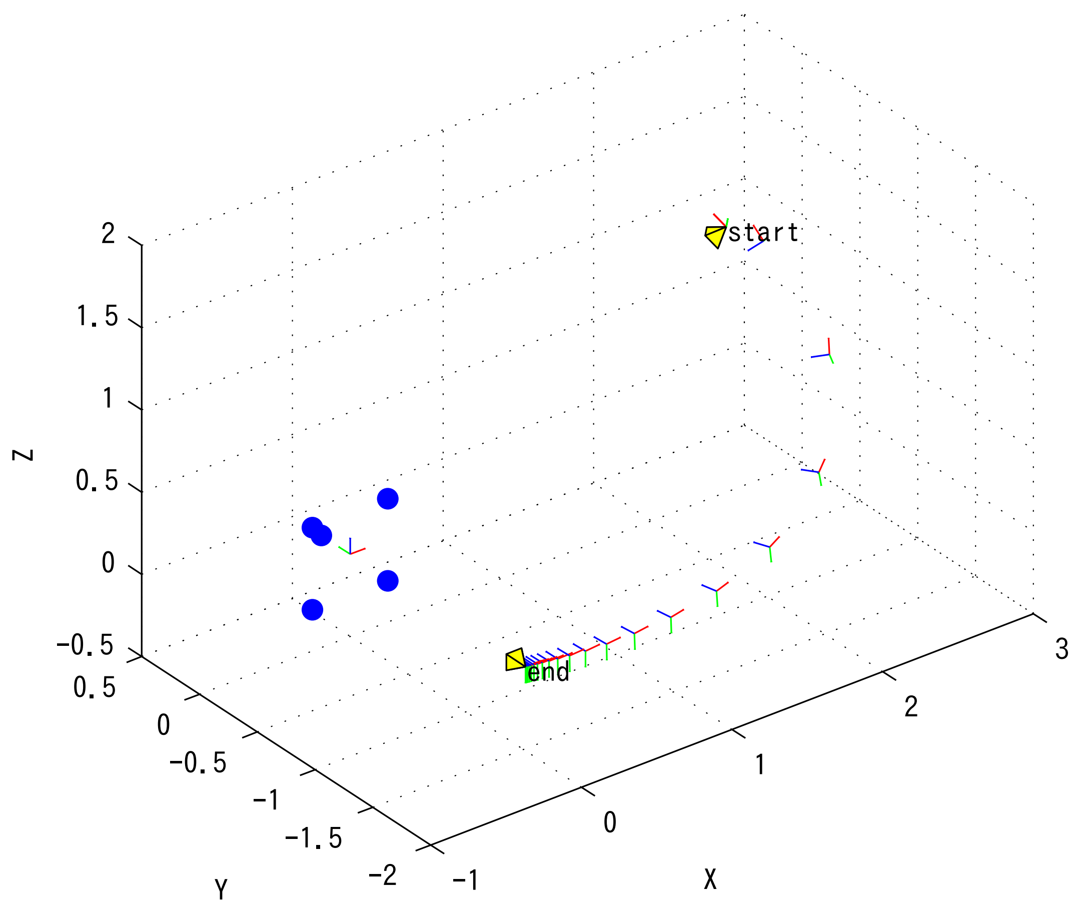


Hybrid visual servo

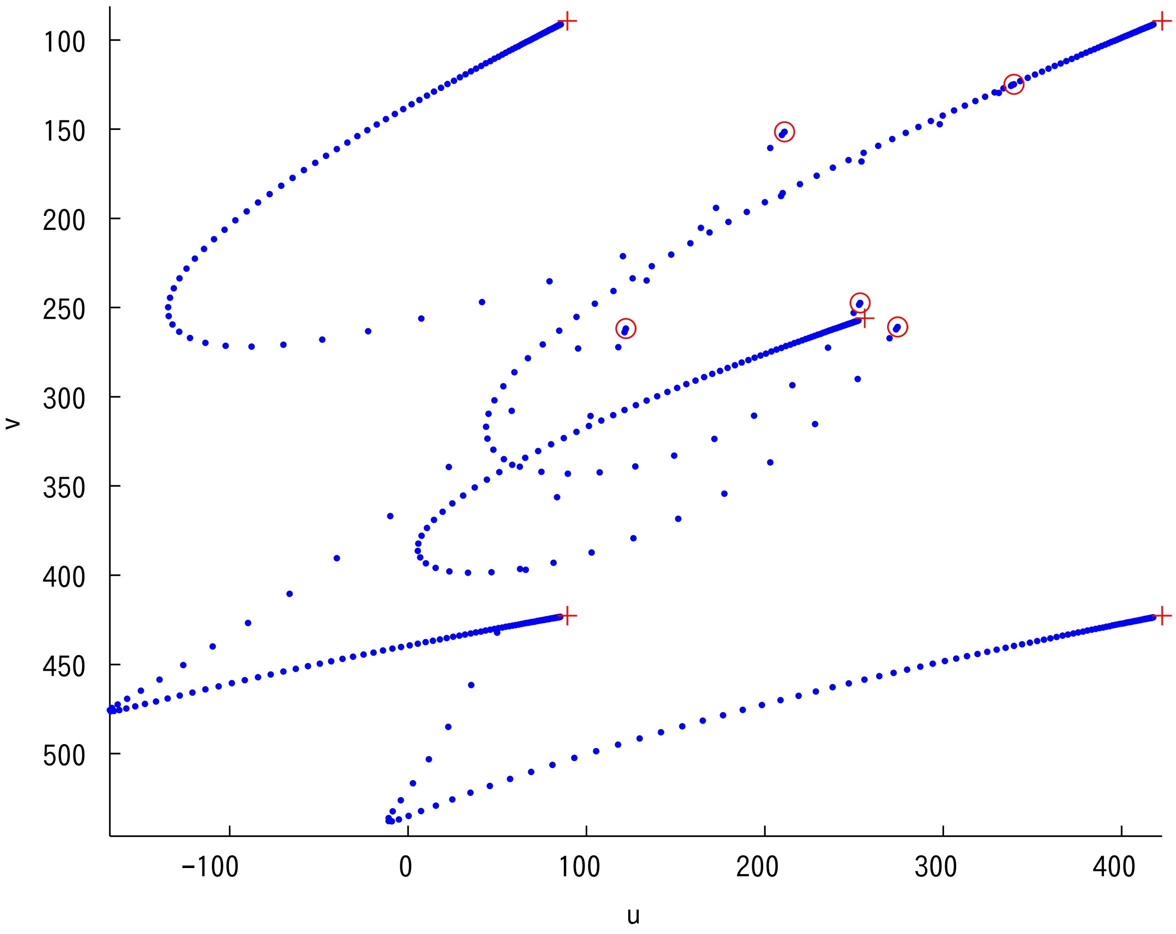
- 2-1/2D visual servo
- Deguchi
- Corke and Hutchinson

- Let the translation error vector be
$$\mathbf{e}_v = \widehat{d}^* \ \mathbf{R}^\top \frac{\mathbf{t}}{d}$$
then this can be computed using Homography estimation.
- Translation error $\mathbf{e}_v$ and angular error $\mathbf{e}_\omega$
$$\dot{\mathbf{s}} = \left[\begin{array}{cc} \mathbf{J}_v & \mathbf{J}_\omega \end{array} \right] \left[\begin{array}{c} \mathbf{e}_v \\ \mathbf{e}_\omega \end{array} \right]$$
- Solving this equation for $\mathbf{e}_\omega$ yields
$$\mathbf{e}_\omega = \mathbf{J}_\omega^{-1}(\dot{\mathbf{s}} - \mathbf{J}_v \mathbf{e}_v)$$
where $\dot{\mathbf{s}}$ should be replaced by $\mathbf{s} - \mathbf{s}^*$.
- Thus, control law becomes
$$\mathbf{v} = -\lambda \left[\begin{array}{c} \mathbf{e}_v \\ \mathbf{e}_\omega \end{array} \right]$$

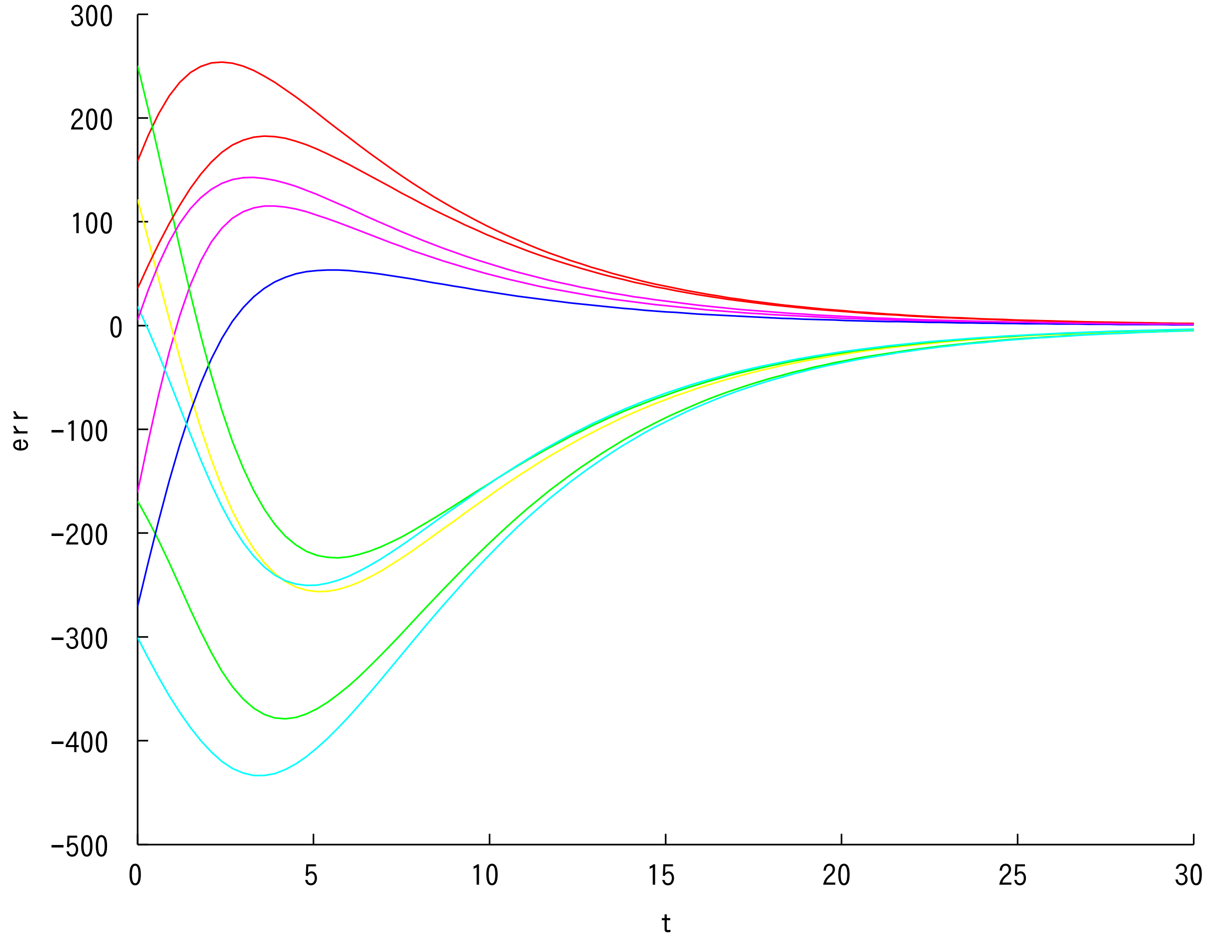
Deguchi: Camera trajectory



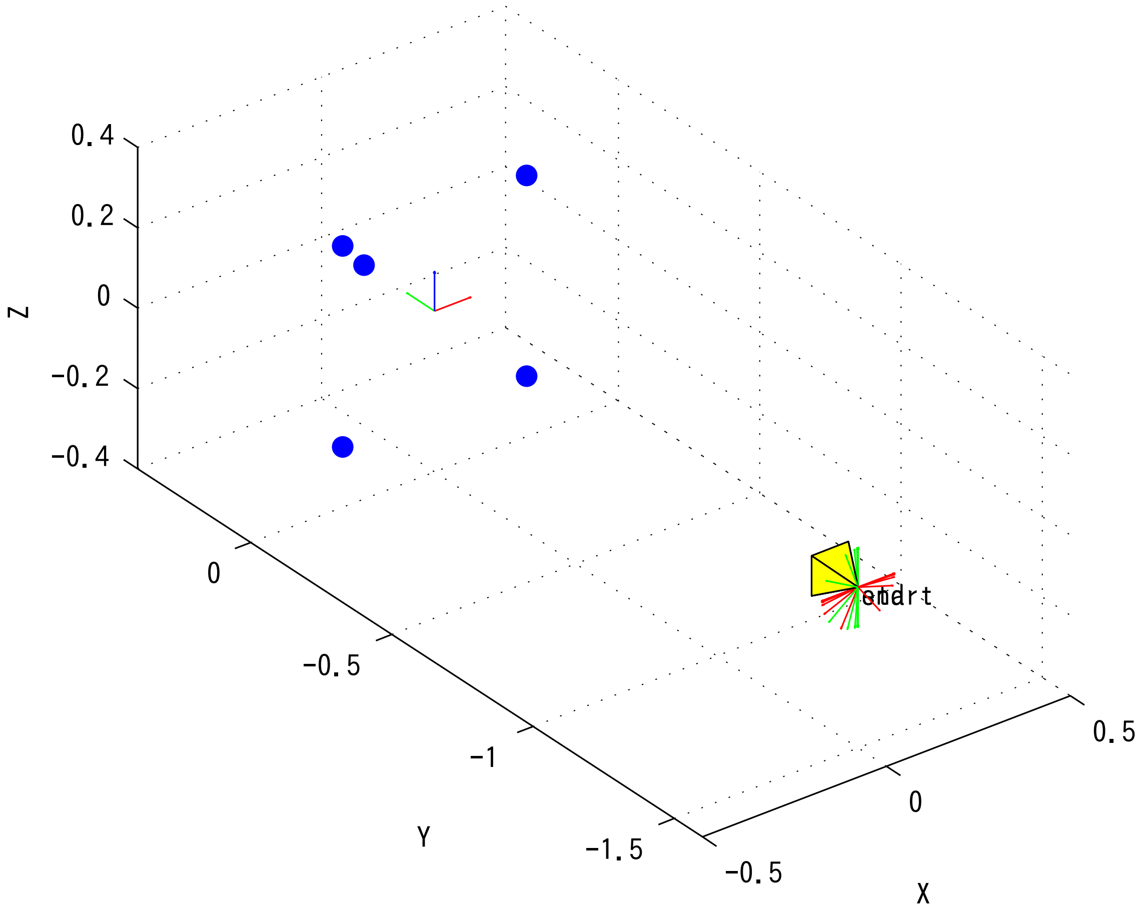
Deguchi: Feature trajectory



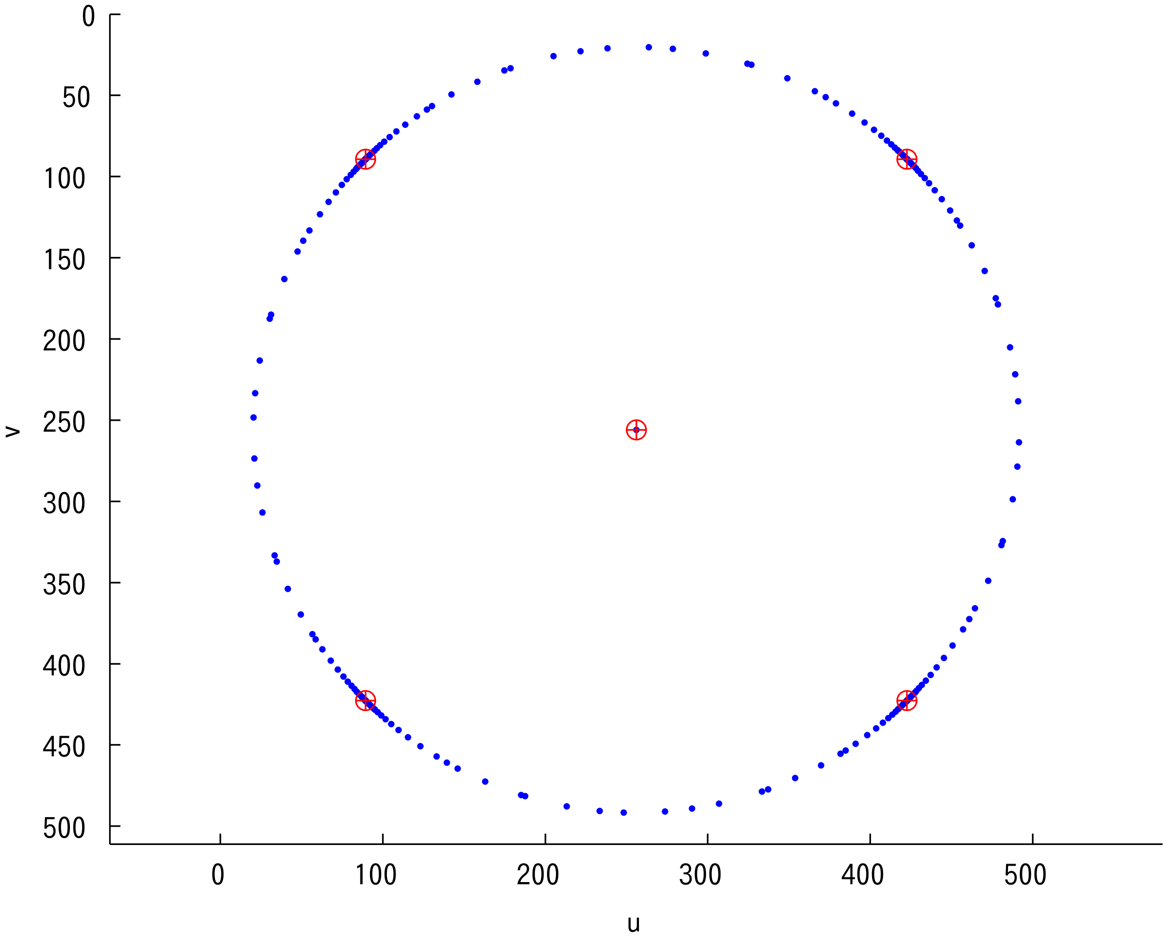
Deguchi: Feature error



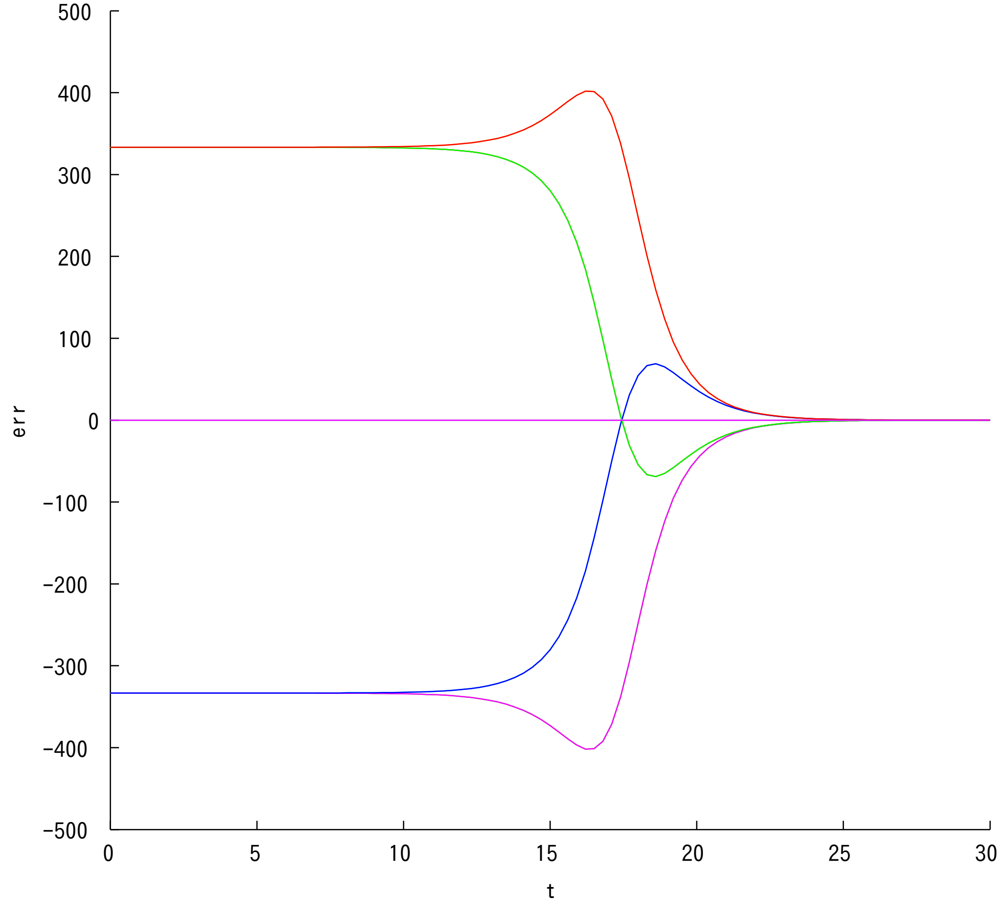
Deguchi symmetric: Camera trajectory



Deguchi symmetric: Feature trajectory

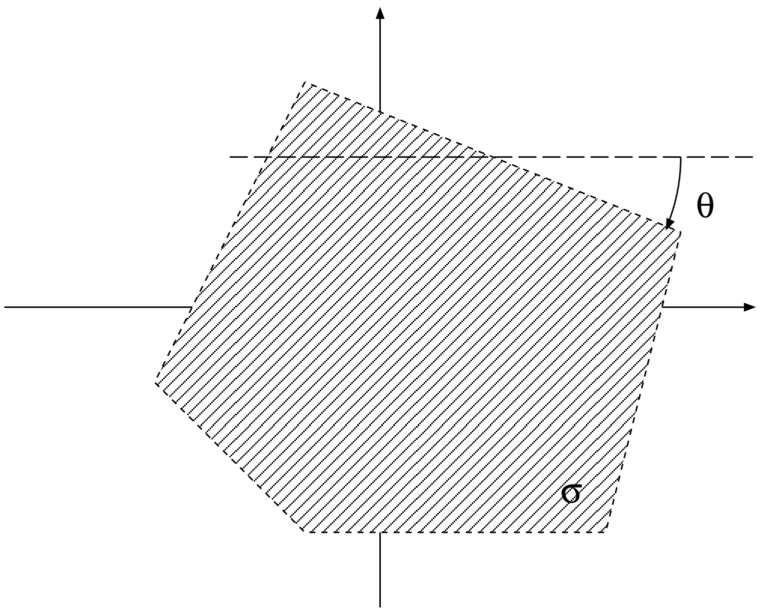


Deguchi symmetric: Feature error



Hybrid visual servo

- 2-1/2D visual servo
- Deguchi
- Corke and Hutchinson



- To control the depth direction use the area σ insde feature points (area of region bounded by feature points).

$$\mathbf{e}_{tz} = -\gamma_T(\sigma - \sigma^*)$$

- Rotation around Z axis is controlled by the angle θ between a line segment connecting two feature points and image horizontal axis.

$$\mathbf{e}_{\omega z} = -\gamma_\omega(\theta - \theta^*)$$

Corke and Hutchinson

- The velocity along Z axis and angular velocity along Z axis of the camera are

$$\mathbf{u}_z = [\mathbf{v}_z \ \boldsymbol{\omega}_z]^\top$$

- The velocities for other DOF are

$$\mathbf{u}_{xy} = [\mathbf{v}_x \ \mathbf{v}_y \ \boldsymbol{\omega}_x \ \boldsymbol{\omega}_y]^\top$$

- Feature velocity is

$$\dot{\mathbf{s}} = \mathbf{J}_{xy}\mathbf{u}_{xy} + \mathbf{J}_z\mathbf{u}_z$$

  where $\mathbf{J}_{xy}$ is the 1, 2, 4, 5-th column and $\mathbf{J}_z$ is the 3, 6-th column of the image Jacobi matrix .

Corke and Hutchinson

- Define the camera velocity concerning Z axis be

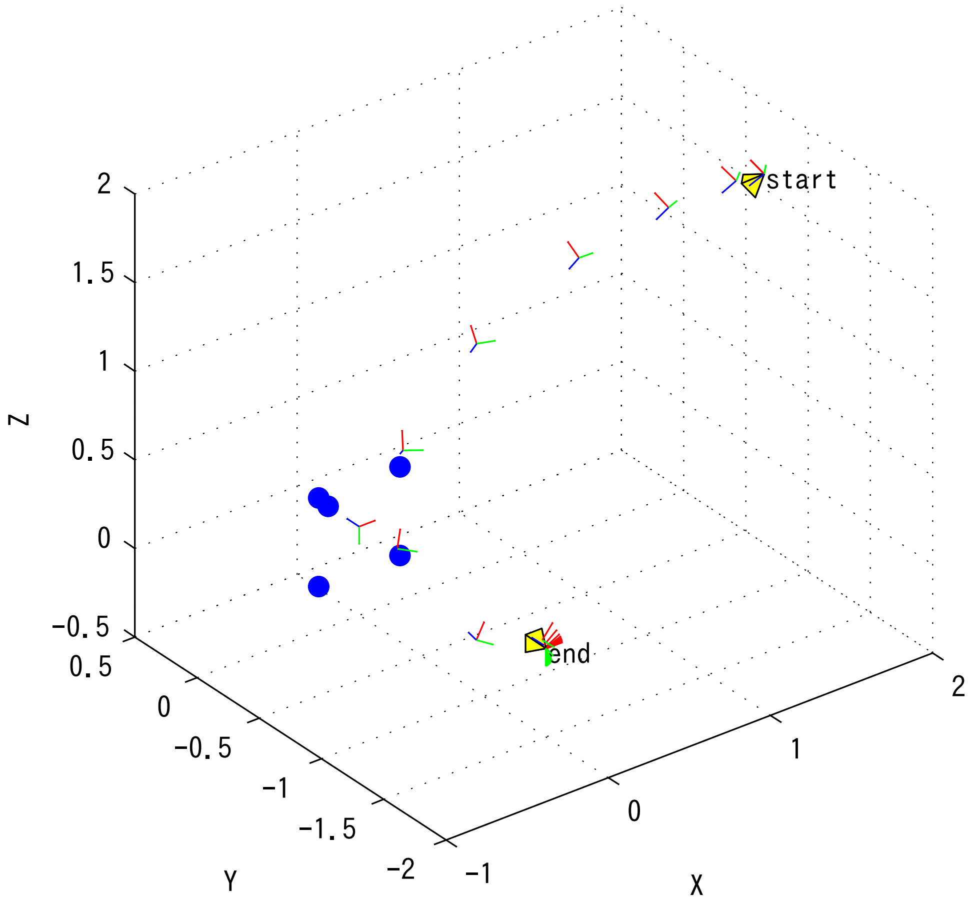
$$\mathbf{u}_z = \left[\begin{array}{c} \mathbf{e}_{tz} \\ \mathbf{e}_{\omega z} \end{array} \right]$$

then we have the velocity for the other DOF as

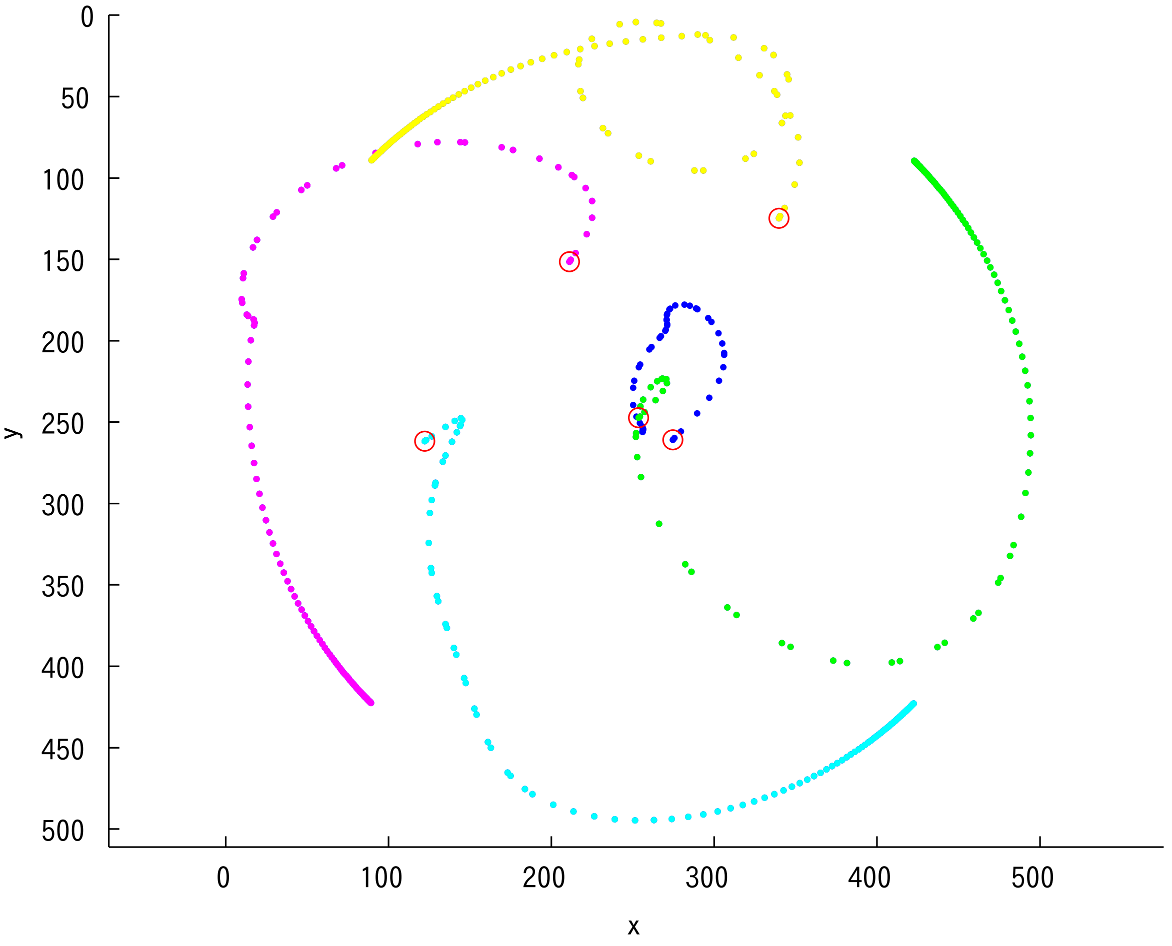
$$\mathbf{u}_{xy} = \mathbf{J}_{xy}^{\dagger}(\dot{\mathbf{s}} - \mathbf{J}_z \mathbf{u}_z)$$

where $\dot{\mathbf{s}}$ should be replaced by $\mathbf{s} - \mathbf{s}^*$.

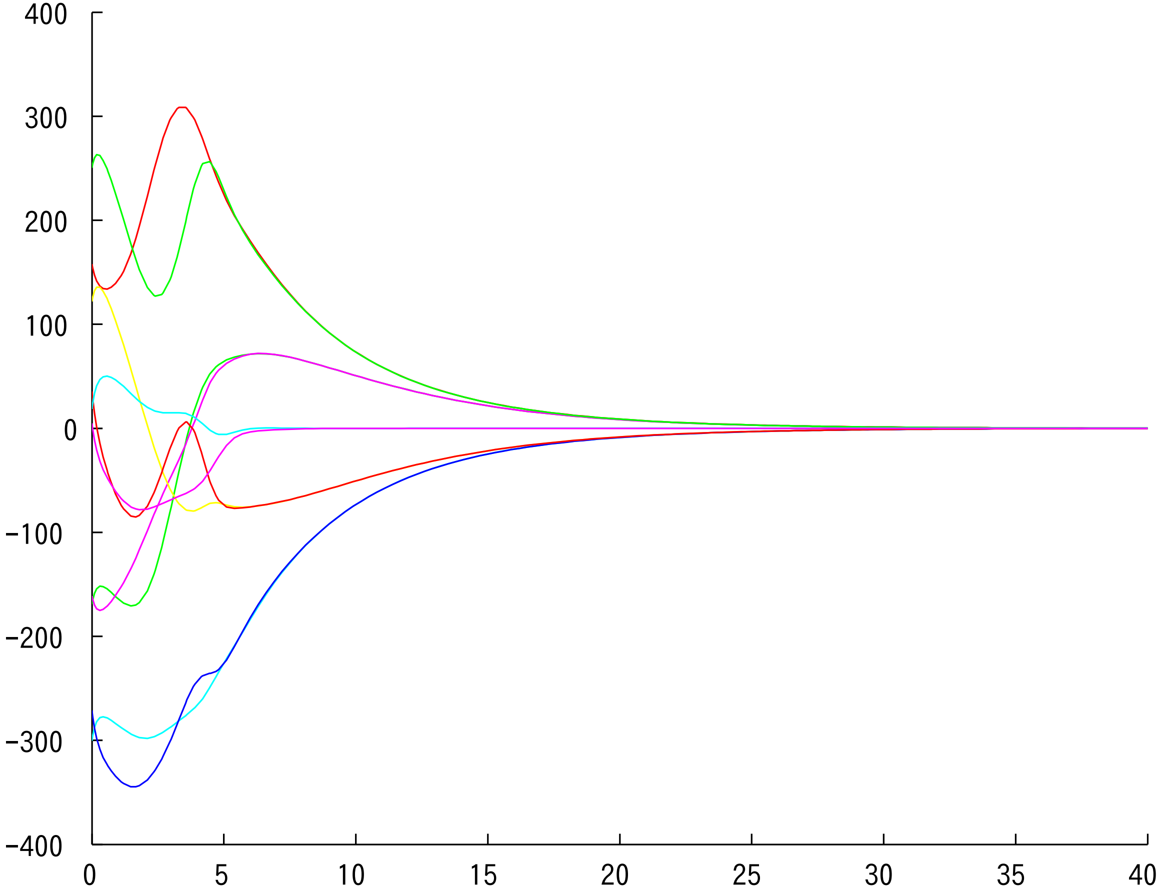
PKSH: Camera trajectory



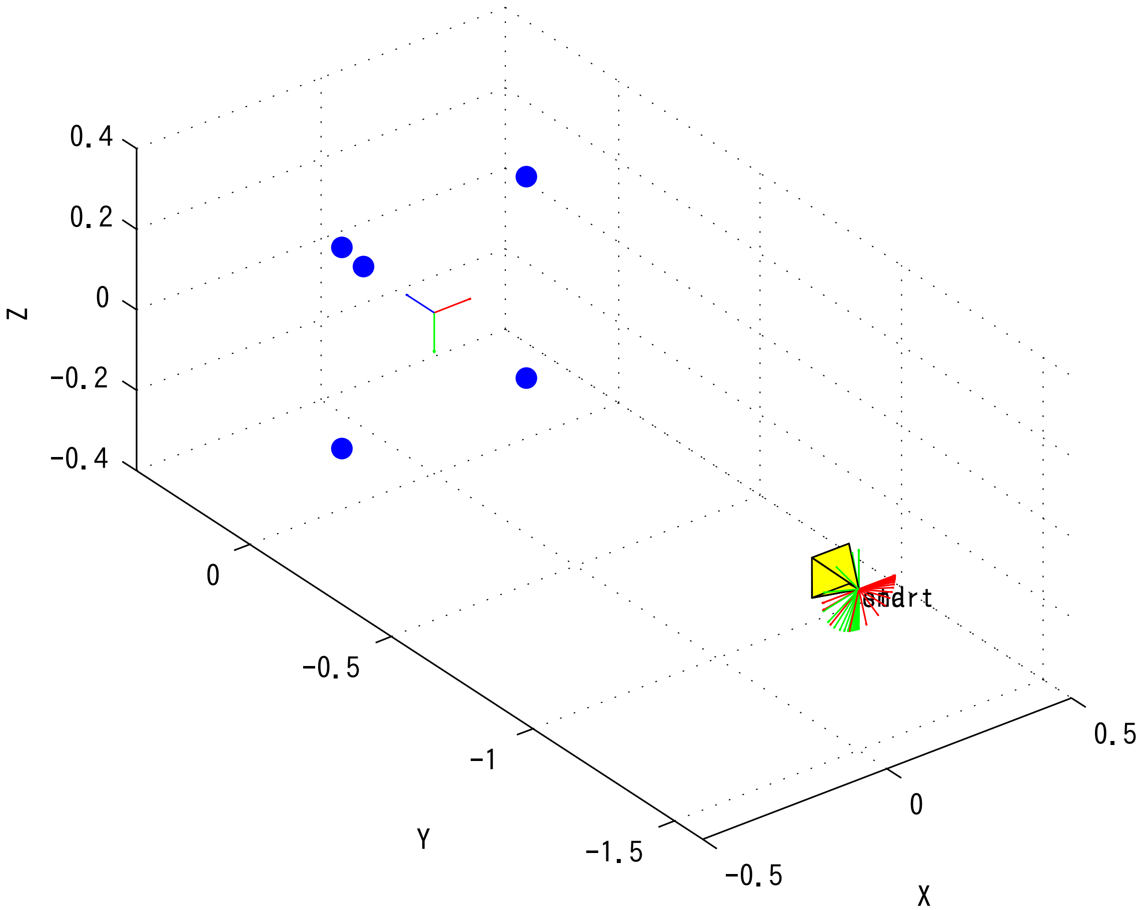
PKSH: Feature trajectory



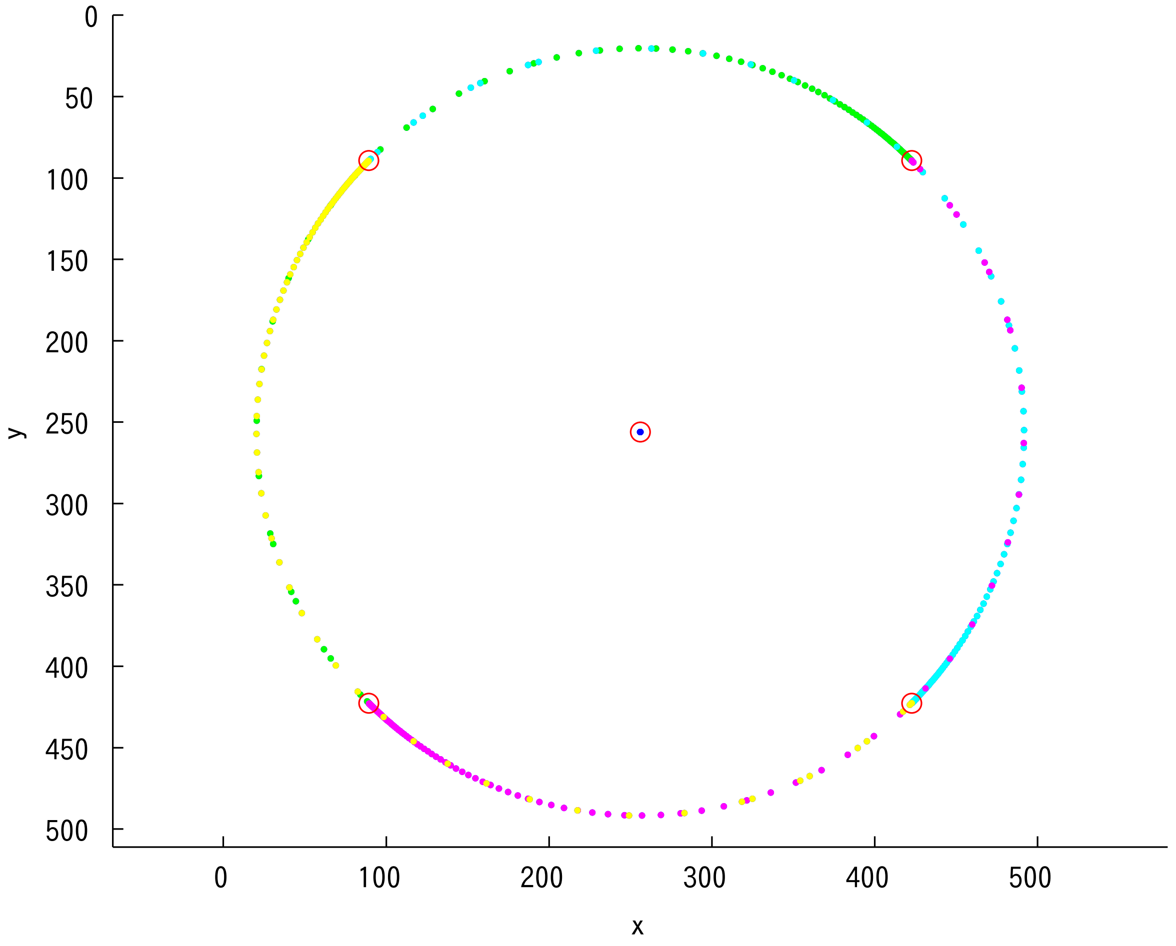
PKSH: Feature error



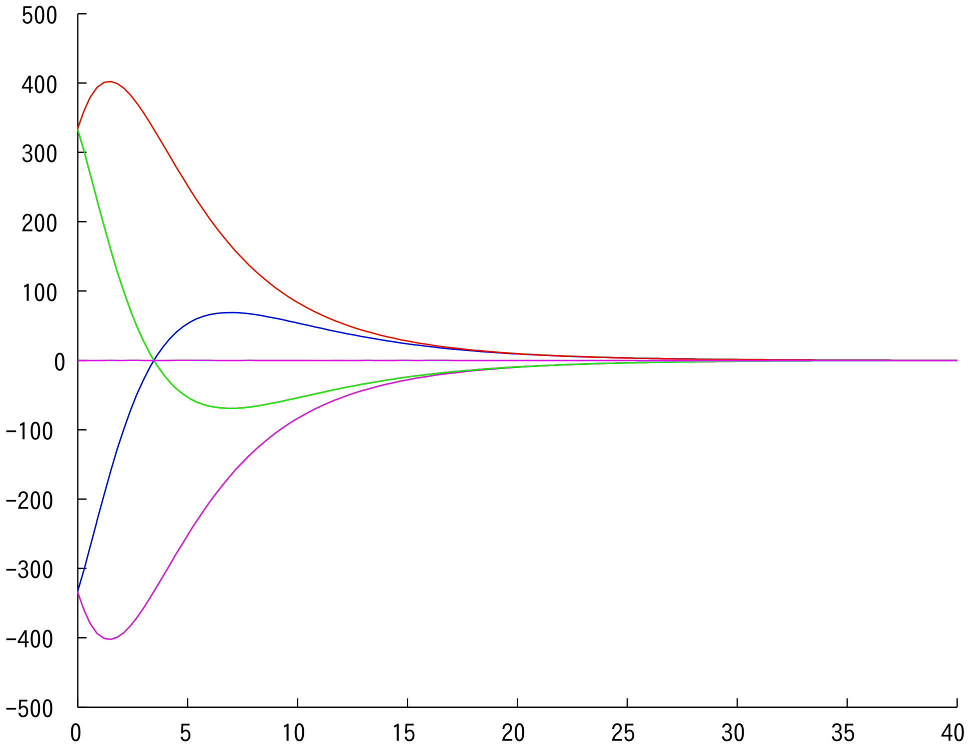
PKSH symmetric: Camera trajectory



PKSH symmetric: Feature trajectory



PKSH symmetric: Feature error



Menu: Course II

- 3D visual servo
- 2D visual servo
- 2.5D visual servo
- ESM algorithm and visual tracking

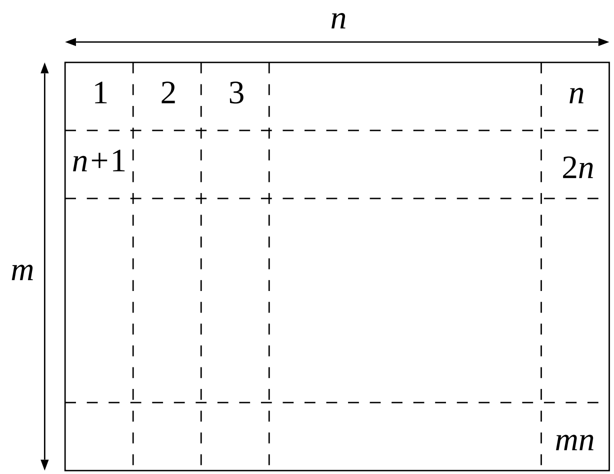
ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm

Image brightness map

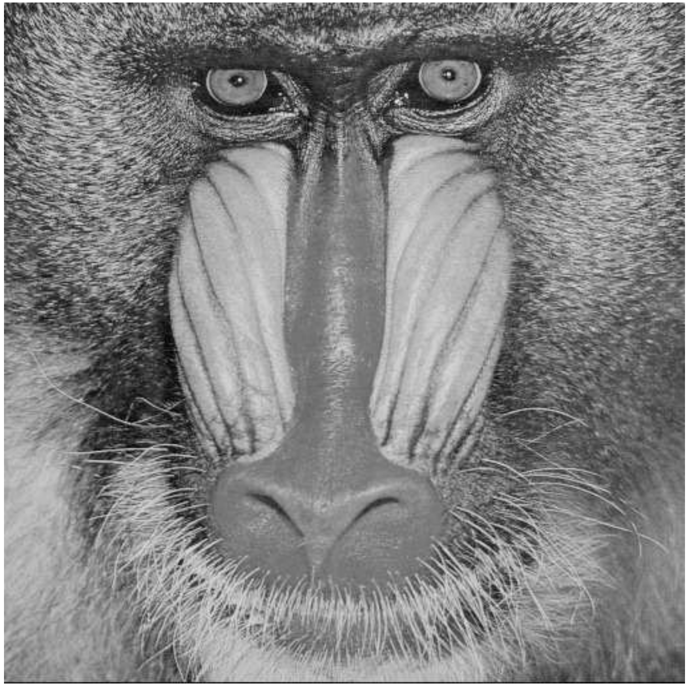


- Brightness pattern of a region
- Image coordinate: $\mathbf{x} = [x\ y\ 1]^\top$
- Brightness of this point: $I(\mathbf{x})$
- Brightness map

$$\mathbf{y} = [I(\mathbf{x}_1)\ \ I(\mathbf{x}_2)\ \ \cdots\ \ I(\mathbf{x}_q)]^\top$$

- Note that x, y may not be integers.

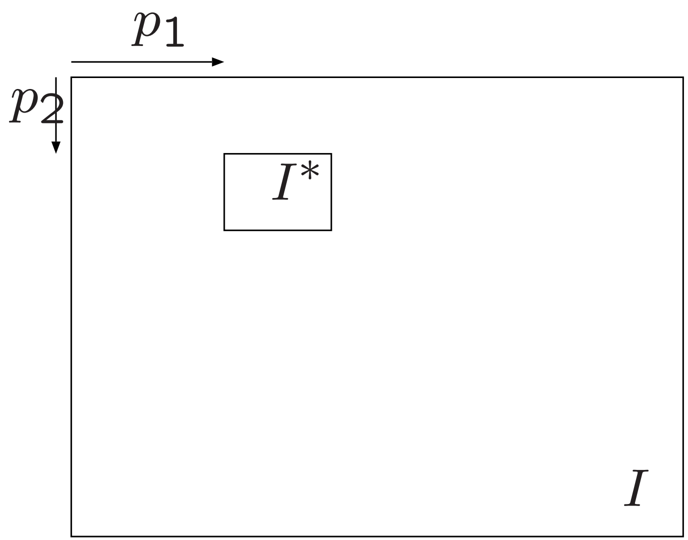
Warp



- **Warp**: How to clip a subregion from brightness map

$$\mathbf{x}' = \mathbf{w}(\mathbf{x}; \mathbf{p})$$

Translation warp



- Warp parametrization: $\mathbf{p} = [p_1, p_2]^\top$

$$\mathbf{w}(\mathbf{x}; \mathbf{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$$

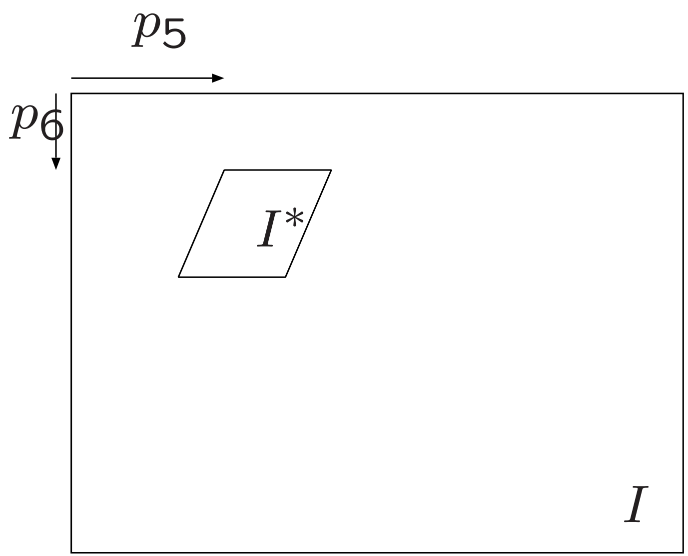
Translation and rotation warp

- Translation p_1, p_2
- Rotatioin p_3

$$\begin{aligned}\mathbf{w}(\mathbf{x};\mathbf{p}) &= \begin{bmatrix} \cos(p_3) & -\sin(p_3) \\ \sin(p_3) & \cos(p_3) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \\ &= \begin{bmatrix} \cos(p_3) & -\sin(p_3) & p_1 \\ \sin(p_3) & \cos(p_3) & p_2 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}\end{aligned}$$

- Obviously the warped coordinates $\mathbf{x}' = \mathbf{w}(\mathbf{x};\mathbf{p})$ are not integer.

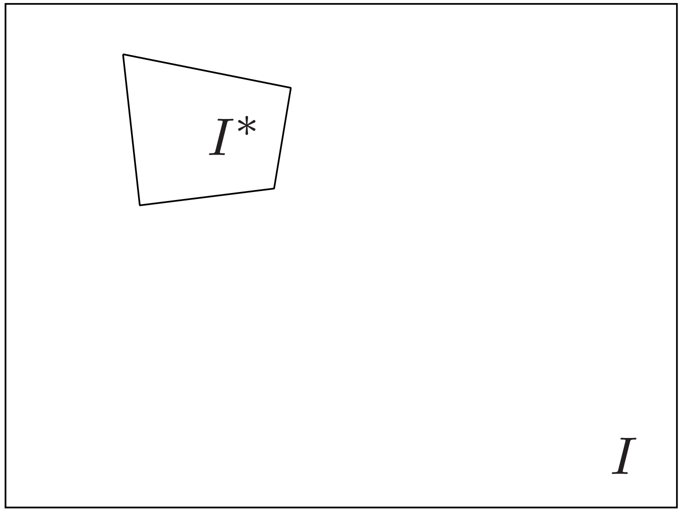
Affine warp



- Warp parameters $\mathbf{p} = [p_1, \ldots, p_6]^\top$

$$\mathbf{w}(\mathbf{x};\mathbf{p}) = \begin{bmatrix} (1+p_1)x + p_3 y + p_5 \\ p_2 x + (1+p_4)y + p_6 \end{bmatrix} = \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homography warp



- Homography is used for planer objects

$$s\mathbf{x} = \mathbf{G}\mathbf{x}^*$$

- Let g_{ij} be the (i, j) element of $\mathbf{G}$ and $g_{33} = 1$.

Homography warp

- Then we have

$$s = g_{31}x^* + g_{32}y^* + 1$$

and

$$\mathbf{x} = \mathbf{w}(\mathbf{G}\mathbf{x}^*) = \begin{bmatrix} \frac{g_{11}x^*+g_{12}y^*+g_{13}}{g_{31}x^*+g_{32}y^*+1} \\ \frac{g_{21}x^*+g_{22}y^*+g_{33}}{g_{31}x^*+g_{32}y^*+1} \\ 1 \end{bmatrix}$$

- Warp parameters $\mathbf{p} = [p_1, \ldots, p_8]^\top$

$$\mathbf{w}(\mathbf{x}; \mathbf{p}) = \begin{bmatrix} \frac{p_1x+p_2y+p_3}{p_7x+p_8y+1} \\ \frac{p_4x+p_5y+p_6}{p_7x+p_8y+1} \\ 1 \end{bmatrix}$$

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm

Template matching

- Template image brightness map

$$\mathbf{y}^* = [I_1^* \quad I_2^* \quad \cdots \quad I_q^*]^\top$$

- **Template matching**: Find $\mathbf{p}$ such that

$$\sum_{\mathbf{x} \in T} \left[I(\mathbf{w}(\mathbf{x}; \mathbf{p})) - I^* \right]^2 \rightarrow \text{min}$$

Template matching as visual servo

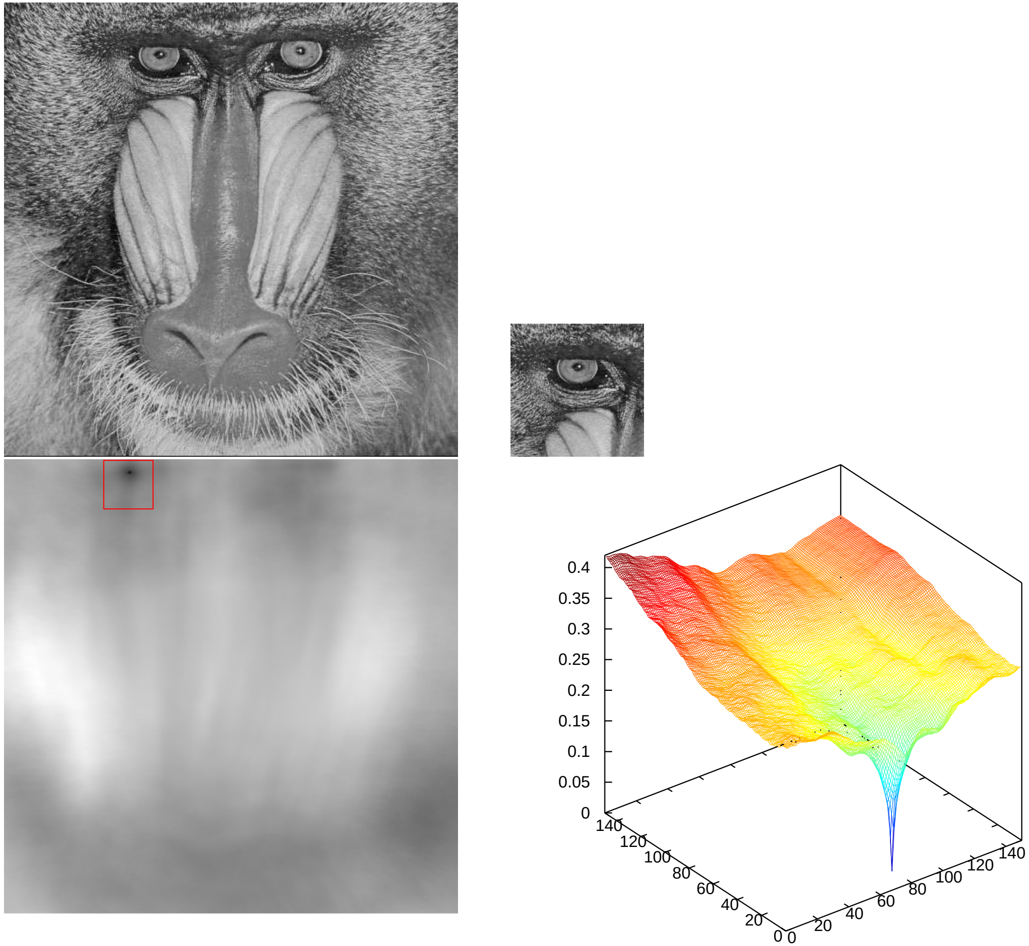
- Suppose

$$\begin{aligned} \mathbf{s}(\mathbf{p}) &= [I(\mathbf{w}(\mathbf{x}_1;\mathbf{p})) \;\; I(\mathbf{w}(\mathbf{x}_2;\mathbf{p})) \;\; \cdots \;\; I(\mathbf{w}(\mathbf{x}_q;\mathbf{p}))]^\top \\ \mathbf{s}^* &= [I_1^* \;\; I_2^* \;\; \cdots \;\; I_q^*]^\top \end{aligned}$$

- Template matching is visual servo in which the error function is defined by

$$\mathbf{e} = \mathbf{s}(\mathbf{p}) - \mathbf{s}^*$$

2D template matching example



ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
  - Additional
  - Compositional
  - Inverse compositional
- ESM algorithm

Lucas-Kanade algorithm formulation

- Suppse we have a estimation of $\mathbf{p}$ and we want to update the parameter by computing $\Delta\mathbf{p}$ by minimizing the cost function

$$\sum_{\mathbf{x}\in T} \left[I(\mathbf{w}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - I^*(\mathbf{x})\right]^2$$

- In other words, suppose that we have $\mathbf{p}$ and want to find $\Delta\mathbf{p}$ that minimize

$$\|\mathbf{e}\| = \|\mathbf{s}(\mathbf{p}+\Delta\mathbf{p}) - \mathbf{s}^*\|$$

and updete the parameter by

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$$

- An iteretion will stop when $\|\Delta\mathbf{p}\| < \epsilon$.

Lucas-Kanade algorithm derivation

- Taylor expansion of $I(\mathbf{w}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p}))$ at $\Delta\mathbf{p}=\mathbf{0}$

$$I(\mathbf{w}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) = I(\mathbf{w}(\mathbf{x};\mathbf{p})) + \nabla I \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \Delta\mathbf{p}$$

- In other words

$$\mathbf{s}(\mathbf{p}+\Delta\mathbf{p}) = \mathbf{s}(\mathbf{p}) + \mathbf{J}(\mathbf{p})\Delta\mathbf{p}$$

where

$$\begin{aligned}
\mathbf{J}(\mathbf{p}) &= \mathbf{J}_I(\mathbf{p})\mathbf{J}_\mathbf{w} = \begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_q \end{bmatrix} \frac{\partial \mathbf{w}}{\partial \mathbf{p}}, \\
\mathbf{J}_I(\mathbf{p}) &= \begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_q \end{bmatrix}, \quad \mathbf{J}_\mathbf{w} = \frac{\partial \mathbf{w}}{\partial \mathbf{p}}
\end{aligned}$$

Lucas-Kanade algorithm derivation

- In this equation

$$\nabla I_k = \nabla I|_{\mathbf{w}(\mathbf{x}_k;\mathbf{p})} = \left[\begin{array}{cc} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{array} \right]\Big|_{\mathbf{w}(\mathbf{x}_k;\mathbf{p})}$$

is the gradient of I evaluated at the warped point $\mathbf{w}(\mathbf{x}_k;\mathbf{p})$.

- While

$$\frac{\partial \mathbf{w}}{\partial \mathbf{p}} = \left[\begin{array}{cccc} \frac{\partial w_x}{\partial p_1} & \frac{\partial w_x}{\partial p_2} & \cdots & \frac{\partial w_x}{\partial p_n} \\ \frac{\partial w_y}{\partial p_1} & \frac{\partial w_y}{\partial p_2} & \cdots & \frac{\partial w_y}{\partial p_n} \end{array} \right]$$

where

$$\mathbf{w}(\mathbf{x};\mathbf{p}) = \left[\begin{array}{c} w_x(\mathbf{x};\mathbf{p}) \\ w_y(\mathbf{x};\mathbf{p}) \end{array} \right]$$

Lucas-Kanade algorithm derivation

- The cost function to be minimized

$$\|\mathbf{s}(\mathbf{p}) + \mathbf{J}(\mathbf{p})\Delta\mathbf{p} - \mathbf{s}^*\|$$

- Nonlinear minimization

$$\Delta\mathbf{p} = -\mathbf{S}^{-1}\mathbf{J}^\top(\mathbf{p})\left(\mathbf{s}(\mathbf{p}) - \mathbf{s}^*\right)$$

  where

  - SDM

$$\mathbf{S} = \mathbf{I}$$

  - GNM

$$\mathbf{S} = \mathbf{J}^\top(\mathbf{p})\mathbf{J}(\mathbf{p})$$

  - LMM

$$\mathbf{S} = \mathbf{J}^\top(\mathbf{p})\mathbf{J}(\mathbf{p}) + \gamma\mathbf{D}$$

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
  - Additional
  - Compositional
  - Inverse compositional
- ESM algorithm

Lucas-Kanade compositional algorithm

- Compositional warp

$$I(\mathbf{w}(\mathbf{w}(\mathbf{x};\Delta\mathbf{p});\mathbf{p})) - I^*(\mathbf{x})$$

- Warp update with warp increment: $\mathbf{w}(\mathbf{x};\Delta\mathbf{p})$

$$\mathbf{w}(\mathbf{x};\mathbf{p}) \leftarrow \mathbf{w}(\mathbf{x};\mathbf{p}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{p}) = \mathbf{w}(\mathbf{w}(\mathbf{x};\Delta\mathbf{p});\mathbf{p})$$

Compositional warp: translation

- Translation warp

$$\mathbf{w}(\mathbf{x};\mathbf{p}') = \mathbf{w}(\mathbf{x};\mathbf{p}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{p}) = \begin{bmatrix} x + p_1 + \Delta p_1 \\ y + p_2 + \Delta p_2 \end{bmatrix}$$

- Warp parameters

$$\begin{bmatrix} p'_1 \\ p'_2 \end{bmatrix} = \begin{bmatrix} p_1 + \Delta p_1 \\ p_2 + \Delta p_2 \end{bmatrix}$$

Compositional warp: Affine

- Affine warp

$$
\begin{aligned}
\mathbf{w}(\mathbf{x};\mathbf{p}') &= \mathbf{w}(\mathbf{x};\mathbf{p}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{p}) \\
&= \begin{bmatrix} 1+p_1 & p_3 & p_5 \\ p_2 & 1+p_4 & p_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1+\Delta p_1 & \Delta p_3 & \Delta p_5 \\ \Delta p_2 & 1+\Delta p_4 & \Delta p_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 1+p'_1 & p'_3 & p'_5 \\ p'_2 & 1+p'_4 & p'_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
\end{aligned}
$$

- Warp parameters

$$
\begin{bmatrix} p'_1 \\ p'_2 \\ p'_3 \\ p'_4 \\ p'_5 \\ p'_6 \end{bmatrix} = \begin{bmatrix} p_1 + \Delta p_1 + p_1\Delta p_1 + p_3\Delta p_2 \\ p_2 + \Delta p_2 + p_2\Delta p_1 + p_4\Delta p_2 \\ p_3 + \Delta p_3 + p_1\Delta p_3 + p_3\Delta p_4 \\ p_4 + \Delta p_4 + p_2\Delta p_3 + p_4\Delta p_4 \\ p_5 + \Delta p_5 + p_1\Delta p_5 + p_3\Delta p_6 \\ p_6 + \Delta p_6 + p_2\Delta p_5 + p_4\Delta p_6 \end{bmatrix}
$$

Compositional warp: Homography

- Homography warp

$$
\begin{aligned}
\mathbf{w}(\mathbf{x};\mathbf{p}') &= \mathbf{w}(\mathbf{x};\mathbf{p}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{p}) \\
&= \mathbf{w}(\mathbf{G}\mathbf{w}(\Delta\mathbf{G}\mathbf{x})) \\
&= \mathbf{w}\left(\begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & 1 \end{bmatrix} \begin{bmatrix} \frac{\Delta p_1 x + \Delta p_2 y + \Delta p_3}{\Delta p_7 x + \Delta p_8 y + 1} \\ \frac{\Delta p_4 x + \Delta p_5 y + \Delta p_6}{\Delta p_7 x + \Delta p_8 y + 1} \\ 1 \end{bmatrix} \right) \\
&= \begin{bmatrix} \frac{p'_1 x + p'_2 y + p'_3}{p'_7 x + p'_8 y + 1} \\ \frac{p'_4 x + p'_5 y + p'_6}{p'_7 x + p'_8 y + 1} \\ 1 \end{bmatrix}
\end{aligned}
$$

Compositional warp: Homography

- Warp parameters

$$
\begin{bmatrix} p_1' \\ p_2' \\ p_3' \\ p_4' \\ p_5' \\ p_6' \\ p_7' \\ p_8' \end{bmatrix}
= \frac{1}{q}
\begin{bmatrix}
p_1 \Delta p_1 + p_2 \Delta p_4 + p_3 \Delta p_7 \\
p_1 \Delta p_2 + p_2 \Delta p_5 + p_3 \Delta p_8 \\
p_1 \Delta p_3 + p_2 \Delta p_6 + p_3 \\
p_4 \Delta p_1 + p_5 \Delta p_4 + p_6 \Delta p_7 \\
p_4 \Delta p_2 + p_5 \Delta p_5 + p_6 \Delta p_8 \\
p_4 \Delta p_3 + p_5 \Delta p_6 + p_6 \\
p_7 \Delta p_1 + p_8 \Delta p_4 + \Delta p_7 \\
p_7 \Delta p_2 + p_8 \Delta p_5 + \Delta p_8
\end{bmatrix},
$$

$$
q = p_7 \Delta p_3 + p_8 \Delta p_6 + 1
$$

- Composition

$$
\mathbf{w}(\mathbf{x}; \mathbf{p}') = \mathbf{w}(\mathbf{G}\mathbf{w}(\Delta \mathbf{G}\mathbf{x})) = \mathbf{w}((\mathbf{G}\Delta\mathbf{G})\mathbf{x})
$$

Lucas-Kanade compositional algorithm

- The cost function to be minimized

$$\sum_{\mathbf{x}\in T} \left[I(\mathbf{w}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - I^*(\mathbf{x})\right]^2$$

- Taylor expansion around $\Delta\mathbf{p} = \mathbf{0}$

$$I(\mathbf{w}(\mathbf{w}(\mathbf{x};\mathbf{0});\mathbf{p})) + \nabla I(\mathbf{w})\frac{\partial \mathbf{w}}{\partial \mathbf{p}}\Delta\mathbf{p} - I^*(\mathbf{x})$$

  where $I(\mathbf{w}) = I(\mathbf{w}(\mathbf{x};\mathbf{p}))$ is the warped image and $\nabla I(\mathbf{w})$ is the gradient of warped image.

- Note that $\mathbf{w}(\mathbf{x};\mathbf{0})$ is unit warp

$$\mathbf{w}(\mathbf{x};\mathbf{0}) = \mathbf{x}$$

Lucas-Kanade compositional algorithm

- Thus we have

$$I(\mathbf{w}(\mathbf{x};\mathbf{p})) + \nabla I(\mathbf{w})\frac{\partial \mathbf{w}}{\partial \mathbf{p}}\Delta\mathbf{p} - I^*(\mathbf{x})$$

- Let

$$\mathbf{e} = \mathbf{s}(\mathbf{p}) + \mathbf{J}(\mathbf{p})\Delta\mathbf{p} - \mathbf{s}^*$$

- Then the parameter updete is given by

$$\Delta\mathbf{p} = -\mathbf{S}^{-1}\mathbf{J}^\top(\mathbf{p})\left(\mathbf{s}(\mathbf{p}) - \mathbf{s}^*\right)$$

where

$$\mathbf{J}(\mathbf{p}) = \begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_q \end{bmatrix} \frac{\partial \mathbf{w}}{\partial \mathbf{p}}, \qquad \nabla I_k = \nabla I(\mathbf{w}(\mathbf{x}_k;\mathbf{p}))$$

Lucas-Kanade compositional algorithm

- Difference between additional and compositional algorithms
  1. Gradient:
     - Additional: Gradient of input image evaluated at warp position
     - Compositional: Gradient of warp image
  2. Jacobi matrix
     - Additional: $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$ is evaluated at $(\mathbf{x}; \mathbf{p})$
     - Compositional: $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$ is evaluated at $(\mathbf{x}; \mathbf{0})$

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
  - Additional
  - Compositional
  - Inverse compositional
- ESM algorithm

Lucas-Kanade inverse compositional

- Swap the role of input image and template

$$I^*(\mathbf{w}(\mathbf{x};\Delta\mathbf{p})) - I(\mathbf{w}(\mathbf{x};\mathbf{p}))$$

- Warp update

$$\mathbf{w}(\mathbf{x};\mathbf{p}) \leftarrow \mathbf{w}(\mathbf{x};\mathbf{p}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{p})^{-1}$$

- Taylor expansion of I^* at $\Delta\mathbf{p} = \mathbf{0}$

$$I^*(\mathbf{w}(\mathbf{x};\mathbf{0})) + \nabla I^* \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \Delta\mathbf{p} - I(\mathbf{w}(\mathbf{x};\mathbf{p}))$$

- Since $\mathbf{w}(\mathbf{x};\mathbf{0})$ is unit warp, we have

$$I^*(\mathbf{x}) + \nabla I^* \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \Delta\mathbf{p} - I(\mathbf{w}(\mathbf{x};\mathbf{p}))$$

  where ∇I^* is gradient of template and constant vector. Since the Jacobi matrix $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$ is evaluated at $\mathbf{p} = \mathbf{0}$, the matrix can be computed before start tracking.

Lucas-Kanade inverse compositional

- Based on this discussion, the formulation becomes the minimization of

$$\mathbf{s}^* + \mathbf{J}^*\Delta\mathbf{p} - \mathbf{s}(\mathbf{p})$$

where

$$\mathbf{J}^* = \begin{bmatrix} \nabla I_1^* \\ \nabla I_2^* \\ \vdots \\ \nabla I_q^* \end{bmatrix} \frac{\partial \mathbf{w}}{\partial \mathbf{p}}\bigg|_{\mathbf{p}=\mathbf{0}}, \qquad \nabla I_k^* = \nabla I^*(\mathbf{x}_k)$$

is a constant matrix.

- Then the parameter update is given by

$$\Delta\mathbf{p} = -\mathbf{S}^{-1}\mathbf{J}^{*\top}\left(\mathbf{s}(\mathbf{p}) - \mathbf{s}^*\right)$$

where $\mathbf{S}$ can be selected from SDM, NM, GNM, and LMM.

Lucas-Kanade compositional algorithm

- Note that $\mathbf{J}^*$ and $\mathbf{S}$ can be computed befor start tracking.
- The warp increment $\mathbf{w}(\mathbf{x}; \Delta\mathbf{p})$ should be invertible.

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm
  - Parametrization of homography matrix
  - ESM Formulation
  - ESM Derivation
  - ESM Tracking experiments

Parametrization of homography matrix

- Homography matrix $\mathbf{G}$ has 9 elements and 1 constraint
- Assume $\det \mathbf{G} = 1$ to avoid singularity condition
  - Singularity: When $\det \mathbf{G} = 0$, the object plane is parallel to optical axis.
  - For a matrix $\mathbf{A}$ with trace$\mathbf{A} = 0$, $\mathbf{G} = \exp(\mathbf{A})$ satisfies $\det \mathbf{G} = 0$.
- Parametrization: $\mathbf{z} = [z_1, \ldots, z_8]^\top$

$$\mathbf{G}(\mathbf{z}) = \exp(\mathbf{A}(\mathbf{z})), \quad \mathbf{A}(\mathbf{z}) = \sum_{i=1}^{8} z_i \mathbf{A}_i$$

where $\mathbf{A}_i$ are 8 bases of trace$\mathbf{A} = 0$.

Parametrization of homography matrix

- 8 bases of trace$\mathbf{A} = 0$

$$\mathbf{A}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{A}_3 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{A}_4 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{A}_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{A}_6 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{A}_7 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{A}_8 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm
  - Parametrization of homography matrix
  - ESM Formulation
  - ESM Derivation
  - ESM Tracking experiments

ESM Formulation

- Use compositional warp
- Suppose $\mathbf{z}$ is current parameter and $\Delta\mathbf{z}$ is parameter update, then

$$I(\mathbf{w}(\mathbf{w}(\mathbf{x};\Delta\mathbf{z});\mathbf{z})) - I^*(\mathbf{x})$$

  and warp update is

$$\mathbf{w}(\mathbf{x};\mathbf{z}) \leftarrow \mathbf{w}(\mathbf{x};\mathbf{z}) \circ \mathbf{w}(\mathbf{x};\Delta\mathbf{z}) = \mathbf{w}(\mathbf{w}(\mathbf{x};\Delta\mathbf{z});\mathbf{z})$$

- Taylor expansion of I at $\Delta\mathbf{z} = \mathbf{0}$ is given by

$$I(\mathbf{w}(\mathbf{w}(\mathbf{x};\mathbf{0});\mathbf{z})) + \nabla I(\mathbf{w})\frac{\partial \mathbf{w}}{\partial \mathbf{z}}\Delta\mathbf{z} - I^*(\mathbf{x})$$

- Since $\mathbf{w}(\mathbf{x};\mathbf{0})$ is unit warp, we have

$$I(\mathbf{w}(\mathbf{x};\mathbf{z})) + \nabla I(\mathbf{w})\frac{\partial \mathbf{w}}{\partial \mathbf{z}}\Delta\mathbf{z} - I^*(\mathbf{x})$$

ESM Formulation

- $I(\mathbf{w}) = I(\mathbf{w}(\mathbf{x};\mathbf{z}))$
- $\nabla I(\mathbf{w})$ is gradient of warped image
- Let

$$\begin{aligned}
\mathbf{s}(\mathbf{z}) &= [I(\mathbf{w}(\mathbf{x}_1;\mathbf{z})) \;\; I(\mathbf{w}(\mathbf{x}_2;\mathbf{z})) \;\; \cdots \;\; I(\mathbf{w}(\mathbf{x}_q;\mathbf{z}))]^\top \\
\mathbf{s}^* &= [I^*(\mathbf{x}_1) \;\; I^*(\mathbf{x}_2) \;\; \cdots \;\; I^*(\mathbf{x}_q)]^\top
\end{aligned}$$

then the function to be minimized is

$$\mathbf{e} = \mathbf{s}(\mathbf{z}) - \mathbf{s}^*$$

ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm
  - Parametrization of homography matrix
  - ESM Formulation
  - ESM Derivation
  - ESM Tracking experiments

ESM Derivation

- ESM method use Jacobi matrices at current and desired points.
- Current Jacobi matrix is

$$
\begin{aligned}
\mathbf{J}(\mathbf{z}) &= \mathbf{J}_I(\mathbf{z})\mathbf{J}_{\mathbf{w}}\mathbf{J}_{\mathbf{G}} = \begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_q \end{bmatrix} \frac{\partial \mathbf{w}}{\partial \mathbf{g}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}, \\
\mathbf{J}_I(\mathbf{z}) &= \begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_q \end{bmatrix}, \quad \mathbf{J}_{\mathbf{w}} = \frac{\partial \mathbf{w}}{\partial \mathbf{g}}, \quad \mathbf{J}_{\mathbf{G}} = \frac{\partial \mathbf{g}}{\partial \mathbf{z}}
\end{aligned}
$$

Detail of Jacobi matrix

- Image gradient

$$\nabla I_k = \nabla I(\mathbf{w}(\mathbf{x}_k; \mathbf{p}))$$

- Warp

$$\mathbf{w}(\mathbf{G}\mathbf{x}^*) = \begin{bmatrix} \dfrac{g_{11}x^* + g_{12}y^* + g_{13}}{g_{31}x^* + g_{32}y^* + g_{33}} \\ \dfrac{g_{21}x^* + g_{22}y^* + g_{23}}{g_{31}x^* + g_{32}y^* + g_{33}} \\ 1 \end{bmatrix}$$

- Second Jacobi matrix

$$\mathbf{J}_{\mathbf{w}} = \begin{bmatrix} x^* & y^* & 1 & 0 & 0 & 0 & -ux^* & -uy^* & -u \\ 0 & 0 & 0 & x^* & y^* & 1 & -vx^* & -vy^* & -v \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Detail of Jacobi matrix

- Third Jacobi matrix

$$\mathbf{J}_{\mathbf{G}} = \left[\; [\mathbf{A}_1]_v \;\; [\mathbf{A}_2]_v \;\; \cdots [\mathbf{A}_8]_v \; \right]$$

where $[\mathbf{A}_i]_v$ is a vector composed of elements of $\mathbf{A}_i$.

Jacobi matrix at desired state

- When $\mathbf{z}$ reaches the desired state $\mathbf{z}^*$ then the image $I(\mathbf{w}(\mathbf{x};\mathbf{z}^*))$ becomes $\mathbf{I}^*(\mathbf{x})$.
- Then the Jacobi matrix is

$$\mathbf{J}^* = \mathbf{J}_{I^*}\mathbf{J}^*_{\mathbf{w}}\mathbf{J}^*_{\mathbf{G}} = \begin{bmatrix} \nabla I_1^* \\ \nabla I_2^* \\ \vdots \\ \nabla I_q^* \end{bmatrix} \left.\frac{\partial \mathbf{w}}{\partial \mathbf{g}}\right|_{\mathbf{z}^*} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}$$

where $\mathbf{J}_{I^*}$ is the Jacobi matrix of the template image and

$$\mathbf{J}^*_{\mathbf{w}} = \mathbf{J}_{\mathbf{w}}, \quad \mathbf{J}^*_{\mathbf{G}} = \mathbf{J}_{\mathbf{G}}$$

Jacobi matrix at desired state

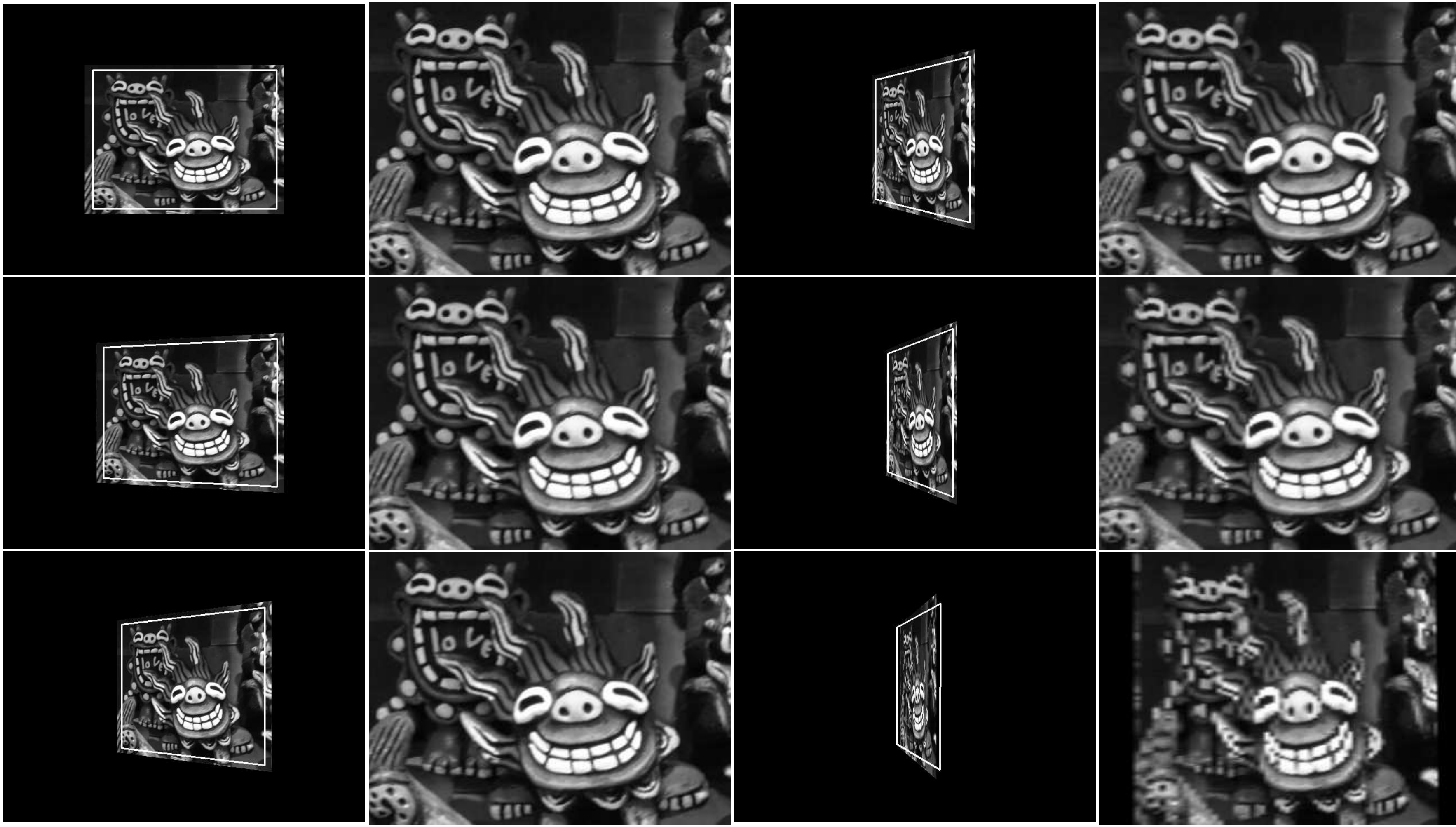
- ESM

$$
\begin{aligned}
\Delta \mathbf{z} &= -\mathbf{J}_{\mathsf{esm}}^{\dagger} \left(\mathbf{s}(\mathbf{z}) - \mathbf{s}^*\right) \\
\mathbf{J}_{\mathsf{esm}} &= \frac{1}{2}(\mathbf{J}(\mathbf{z}) + \mathbf{J}^*) = \frac{1}{2}(\mathbf{J}_I(\mathbf{z}) + \mathbf{J}_{I^*})\mathbf{J}_{\mathbf{w}}\mathbf{J}_{\mathbf{G}}
\end{aligned}
$$

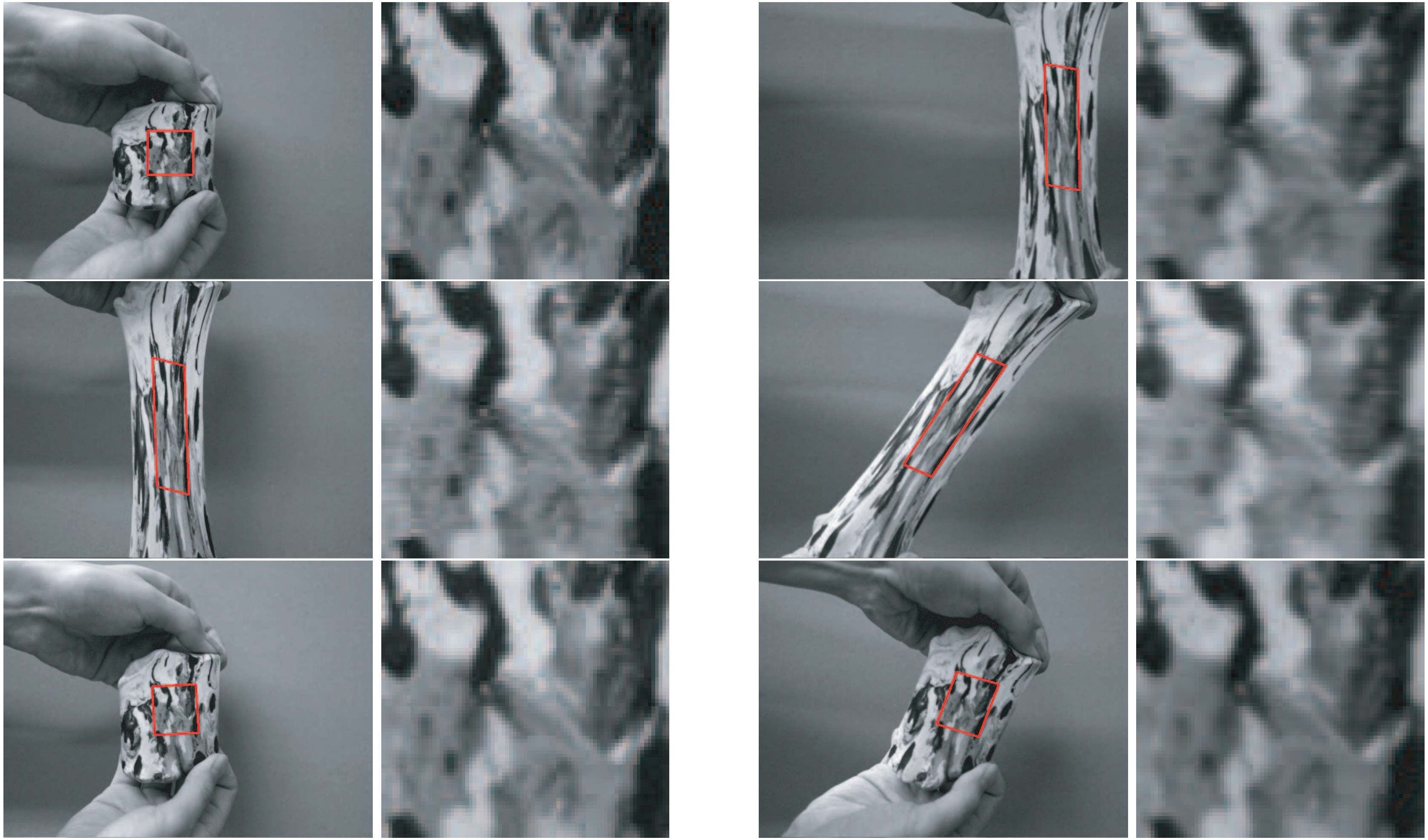
ESM visual tracking

- Image brightness map and warp
- Template matching
- Lucas and Kanade algorithm
- ESM algorithm
  - Parametrization of homography matrix
  - ESM Formulation
  - ESM Derivation
  - ESM Tracking experiments

ESM experimental results - rotation



ESM experimental results - rubber



ESM experimental results - rotation