Automated Visual Inspection Planning by using Stereo Cameras incorporated with Error Analysis

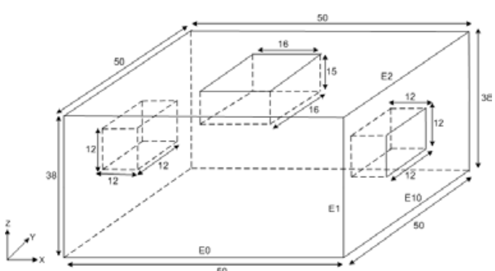
Fai-Lung Shih

Introduction

- How to determine the object satisfies the design specification
- Camera is used as the measurement tool.
- Automated inspection system is developed to generate inspection plans
- Inspection planning is developed to optimize camera poses arrangement

Example Application

Determine a plan to inspect E0, E1, E2, E10.



Visual inspection planning

- How to locate the cameras
  - A set of desired geometric entities
- Location (camera poses)
  - Position (x, y, z)
  - Orientation (ϕ, θ, ψ)
- Plan
  - A set of camera poses

Motivation

- Given an object to inspect, an effective and efficient inspection plan will generate, so that all of desired geometrics entities are accurately measured
- During the process, optimal camera poses are determined.
  - How to define a camera pose is optimal?
  - How to define viewing regions for camera placement so that all the desired geometric entities can be seen?

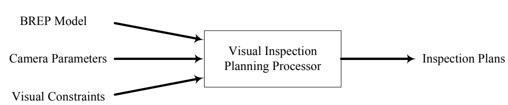
Motivation Cont.

- How to select the best viewpoint within each viewing regions?
- What are the criteria for selecting the camera pose?
- How to select a set of camera poses that will see all the desired entities?
- After optimal camera poses are determined and images are taken, how to measure the entities?

- The automated visual inspection planning system will address all the problems.

Objective

- Develop an automated visual inspection planning system.
- Effective and efficient inspection plans are automatically generated.



Contribution

- An algorithm to convert aspect graph to entity-based aspect graph is implemented.
  - A method to merge viewing domains is introduced.
- A new relative error model is introduced.
- The error model is applied in stereo cameras system.
- Make the visual constraints more complete

Contribution Cont.

- A new nonlinear program is reformulated by integrating with visual constraints, stereo cameras system.
  - An approach to determine the initial feasible camera pose is introduced.
- An automated visual inspection planning system is designed and implemented.
- A simulation system is implemented to verify the inspection plans.

Related Works

- AG, is introduced by Koenderink and van Doorn in 1979.
- Laurentini (1995) introduced reduced aspect graph, which merges the characteristic views (nodes) that have the same line drawings.
- Yang proposed the Entity-based Aspect Graphs (EAGs) to handle potential occlusion in the viewing of desired geometric entities.
- In the MVP system, Tarabanis use a non-iterative approach and five degrees of freedom. Tarabains also redefines the problem as a set of sensor constraints in 8D

Related Works Cont.

- Yang, modeled the displacement error of a camera pose as a function of six degrees of freedom, the camera position (t_x, t_y, t_z) and orientation (Φ, θ, τ).
- Yang also models the quantization error using a uniform distribution in both horizontal and vertical directions.
- Ho models the dimensional errors due to quantization for various geometric entities.
- Griffin approximates the change in the lighting conditions using Gaussian distribution.

Related Works Cont.

- Sakane developed the HEAVEN, Niepold developed the VIO and Yi developed the ICE are using the generate-and-test method.
- GASP is developed by Emanuele Trucco, GASP uses imaging sensors and feature-based object modes for sensor planning.
- Crosby uses nonlinear programming with nonlinear constraints for sensor planning.
  - The mean squared error $E[\in^2]$ is the objective function of the nonlinear program, where $\in$ is the error sum of displacement and quantization errors.

Problem Formulation

$$(\text{BREP}, \text{EOI}, \text{VC}, \text{CP}) \rightarrow \text{IP}$$

$\text{EOI} = \{e_1, e_2, ...\}$

$\text{VC} = \{\text{resolution, focus, field-of-view, veiw-of-angle, visibility}\}$

$\text{CP} = \{\text{focal length, aperature radius, resolutions, image sizes}\}$

$\text{IP} = \{C_i, \text{EOI}_i\}$

Where

BREP : the model of the object

EOI : the set of entities of interest

VC : the set of visual constraints

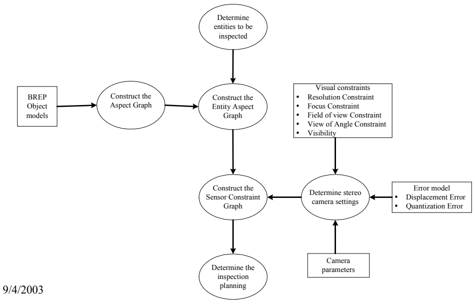
CP : the set of camera parameters

IP : the inspection plan

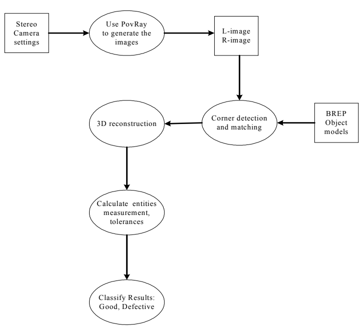
e_i : the set of edges

C_i : the set of cameras' position and orientation

Overview of Inspection Planning System



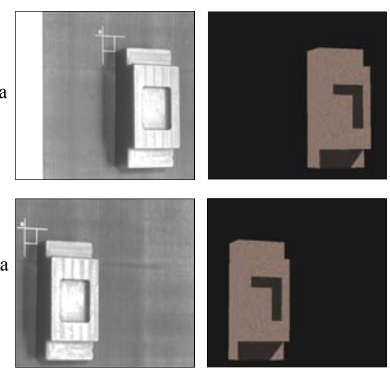
Simulation System



Simulation Example

Left Camera

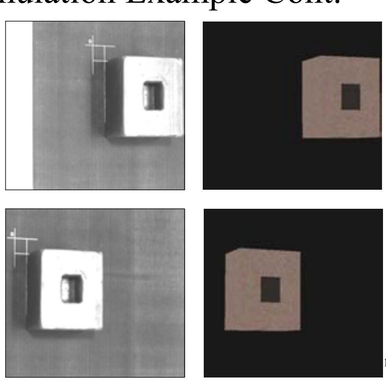
Right Camera



Simulation Example Cont.

Left Camera

Right Camera



Modeling Error

- In robotic vision sensing systems, uncertainty arise due to the following sources:
  - displacement of the camera
  - quantization error in image digitization

Displacement Error ε_d

- difference between the observed dimensions and actual dimensions
- composed of two components,
  - horizontal component (ε_{dx})
  - vertical component (ε_{dy}).
- Errors in camera placement is represented as errors in six degrees of freedom
  - $d_x, d_y, d_z, d_\phi, d_\theta, d_\tau$
- Represented by independent Gaussian variables
  - zero mean
  - variance

Displacement error (*du dv)*

$$du = (\partial \frac{fc_1}{f-c_3} / \partial x)dx + (\partial \frac{fc_1}{f-c_3} / \partial y)dy + (\partial \frac{fc_1}{f-c_3} / \partial z)dz +$$
$$(\partial \frac{fc_1}{f-c_3} / \partial \phi)d\phi + (\partial \frac{fc_1}{f-c_3} / \partial \theta)d\theta + (\partial \frac{fc_1}{f-c_3} / \partial \tau)d\tau$$
$$dv = (\partial \frac{fc_2}{f-c_3} / \partial x)dx + (\partial \frac{fc_2}{f-c_3} / \partial y)dy + (\partial \frac{fc_2}{f-c_3} / \partial z)dz +$$
$$(\partial \frac{fc_2}{f-c_3} / \partial \phi)d\phi + (\partial \frac{fc_2}{f-c_3} / \partial \theta)d\theta + (\partial \frac{fc_2}{f-c_3} / \partial \tau)d\tau$$

$dx, dy, dz, d\phi, d\theta, d\tau$ represents the errors in camera placement

Displacement error (δu δv)

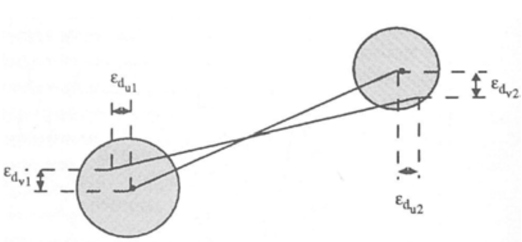
$$\delta_u^2 = (\partial \frac{fc_1}{f-c_3} / \partial x)^2 \delta_x^2 + (\partial \frac{fc_1}{f-c_3} / \partial y)^2 \delta_y^2 + (\partial \frac{fc_1}{f-c_3} / \partial z)^2 \delta_z^2 +$$
$$(\partial \frac{fc_1}{f-c_3} / \partial \phi)^2 \delta_\phi^2 + (\partial \frac{fc_1}{f-c_3} / \partial \theta)^2 \delta_\theta^2 + (\partial \frac{fc_1}{f-c_3} / \partial \tau)^2 \delta_\tau^2$$
$$\delta_v^2 = (\partial \frac{fc_2}{f-c_3} / \partial x)^2 \delta_x^2 + (\partial \frac{fc_2}{f-c_3} / \partial y)^2 \delta_y^2 + (\partial \frac{fc_2}{f-c_3} / \partial z)^2 \delta_z^2 +$$
$$(\partial \frac{fc_2}{f-c_3} / \partial \phi)^2 \delta_\phi^2 + (\partial \frac{fc_2}{f-c_3} / \partial \theta)^2 \delta_\theta^2 + (\partial \frac{fc_2}{f-c_3} / \partial \tau)^2 \delta_\tau^2$$

Displacement error

- ε_d is a Gaussian random variable

$$E[\varepsilon_{dx}] = 0$$
$$E[\varepsilon_{dy}] = 0$$
$$E[\varepsilon_d] = 0$$
$$Var[\varepsilon_{dx}] = 2\delta_u^2$$
$$Var[\varepsilon_{dy}] = 2\delta_v^2$$
$$Var[\varepsilon_d] = \delta_d^2 = 2(\delta_u^2 \cos^2\gamma + \delta_v^2 \sin^2\gamma)$$
$$\gamma = \tan^{-1}((v_1 - v_2)/(u_1 - u_2))$$

Displacement Error ε_d



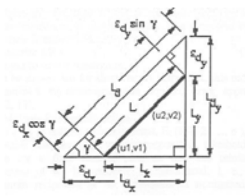
- The horizontal errors, ε_{du1} and ε_{du2}, and the vertical errors ε_{dv1} and ε_{dv2}, due to displacement of the camera

Displacement Error ε_d Cont.

- Let the coordinates of the two endpoints of a line be (u1, v1) and (u2, v2)

$$\varepsilon_{dx} = du1 - du2$$
$$\varepsilon_{dy} = dv1 - dv2$$

Approximate Displacement error



- Dimensional error is geometrically approximated:
  $\varepsilon_d \approx \varepsilon_{dx}\cos(\gamma) + \varepsilon_{dy}\sin(\gamma)$
  γ = angle between line

Displacement error of k lines

$$M_6 = [\eta_{d\psi}\ \eta_{d\theta}\ \eta_{d\phi}\ \eta_{dx}\ \eta_{dy}\ \eta_{dz}]^T$$
$$C_6 = \mathrm{diag}\{\sigma^2_{d\psi}, \sigma^2_{d\theta}, \sigma^2_{d\phi}, \sigma^2_{dx}, \sigma^2_{dy}, \sigma^2_{dz}\}$$

$$M_{div} = [\cdots\ \eta_{\zeta_{ij}}\ \eta_{\xi_{ij}}\ \eta_{\chi_{ij}}\ \cdots]^T$$
$$C_{div} = J_6 C_6 J_6^T$$

$$M_{uv} = J_{div} M_{div}$$
$$C_{uv} = J_{div} C_{div} J_{div}^T$$

$$M_{xy} = J_{uv} M_{uv}$$
$$C_{xy} = J_{uv} C_{uv} J_{uv}^T$$

$$M_d = J_{xy} M_{xy}$$
$$C_d = J_{xy} C_{xy} J_{xy}^T$$

$$J_{div} = \begin{bmatrix} \frac{\partial \varepsilon_{d_u}}{\partial \zeta} & \frac{\partial \varepsilon_{d_u}}{\partial \xi} & \frac{\partial \varepsilon_{d_u}}{\partial \chi} \\ \frac{\partial \varepsilon_{d_v}}{\partial \zeta} & \frac{\partial \varepsilon_{d_v}}{\partial \xi} & \frac{\partial \varepsilon_{d_v}}{\partial \chi} \end{bmatrix}_{\zeta=\eta_\zeta, \xi=\eta_\xi, \chi=\eta_\chi} = \begin{bmatrix} \frac{1}{\eta_\chi} & 0 & -\frac{\eta_\zeta}{\eta_\chi^2} \\ 0 & \frac{1}{\eta_\chi} & -\frac{\eta_\xi}{\eta_\chi^2} \end{bmatrix}$$

$$M_{uv} = \begin{bmatrix} \vdots \\ \eta_{\varepsilon_{d_u}} \\ \eta_{\varepsilon_{d_v}} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ -\frac{\mathrm{Cov}(\zeta,\chi)}{\eta_\chi^2} \\ -\frac{\mathrm{Cov}(\xi,\chi)}{\eta_\chi^2} \\ \vdots \end{bmatrix}$$

$$C_{uv} = \begin{bmatrix} \ddots & & \\ & \sigma^2_{\varepsilon_{d_u}}\ \mathrm{Cov}(\varepsilon_{d_u}, \varepsilon_{d_v}) & \\ & \sigma^2_{\varepsilon_{d_v}} & \\ & & \ddots \end{bmatrix} = J_{div} C_{div} J_{div}^T$$

Displacement error of k lines cont.

- Total dimensional error for k lines is:

$$\varepsilon_d = \sum_{j=1}^{k} \varepsilon_{d_j}$$
$$\eta_{\varepsilon_d} = \sum_{j=1}^{k} M_d(j)$$
$$\sigma^2_{\varepsilon_d} = \sum_{i=1}^{k} \sum_{j=1}^{k} C_d(i,j)$$
$$E[\varepsilon_d^2] = \sigma^2_{\varepsilon_d} + \eta^2_{\varepsilon_d}$$

Quantization Error

- Arise due to image digitization
- Represented by independent uniform variables
  - zero mean
  - variance equal to the pixel resolution

Quantization Error ε_q

- ε_q is a uniform variables

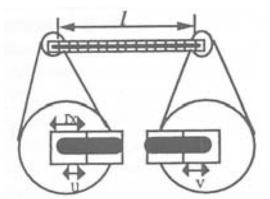
$$E[\varepsilon_{qx}] = 0$$
$$E[\varepsilon_{qy}] = 0$$
$$E[\varepsilon_q] = 0$$
$$Var[\varepsilon_{qx}] = \tfrac{1}{6} r_x^2$$
$$Var[\varepsilon_{qy}] = \tfrac{1}{6} r_y^2$$
$$Var[\varepsilon_q] = \tfrac{1}{6}(r_x^2 \cos^2\gamma + r_y^2 \sin^2\gamma)$$

Quantization error ε_q

- Spatial quantization error
  - 1) one-dimensional
  - 2) two-dimensional
- True length $L = lr_x + u + v$
  - r_x is the width of pixel
  - l is the number of pixels
  - u and v are independent random variables that have a uniform distribution in the range of $[0, r_x]$

After quantization

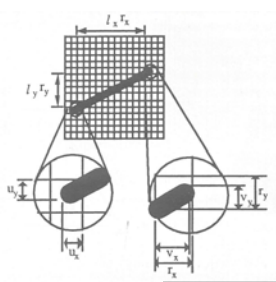
$$L_q = \begin{cases} (l+2)r_x & \text{if } u \ge 0.5r_x \text{ and } \; v \ge 0.5r_x \\ (l+1)r_x & \text{if } (u \ge 0.5r_x \text{ and } \; v < 0.5r_x) \\ & or(u < 0.5r_x \text{ and } \; v \ge 0.5r_x) \\ lr_x & \text{if } u < 0.5r_x \text{ and } \; v < 0.5r_x \end{cases}$$



Two-dimensional quantization errors

- Let r_x and r_y be the width and length of the pixel respectively
- $L_x = l_x r_x + u_x + v_x$
- $L_y = l_y r_y + u_y + v_y$
- Actual length $L = \sqrt{L_x^2 + L_y^2}$
- Quantized length $L_q = \sqrt{L_{qx}^2 + L_{qy}^2}$
- It has four random variables u_x ,v_x , u_y and v_y

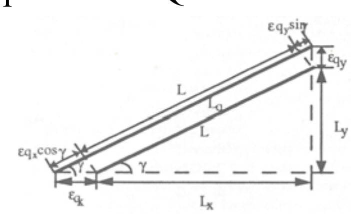
Two-dimensional quantization errors cont.



$L_x = l_x r_x + u_x + v_x$
$L_y = l_y r_y + u_y + v_y$

$$\varepsilon_q = L_q - L = \sqrt{(L\cos\gamma + \varepsilon_{qx})^2 + (L\sin\gamma + \varepsilon_{qy})^2} - L$$

Approximate Quantization error



- if the original line and quantized line are parallel
- $\varepsilon_q \approx \varepsilon_{qx}\cos(\gamma) + \varepsilon_{qy}\sin(\gamma)$

Total quantization error for k lines

- Total dimensional error due to quantization in all lines:

$$\varepsilon_q = \sum_{j=1}^{k} \varepsilon_{q_j}$$

$$\eta_{\varepsilon_{q_j}} \approx 0$$

$$\sigma^2_{\varepsilon_{q_j}} \approx \tfrac{1}{6}(r_x^2 \cos^2\gamma + r_y^2 \sin^2\gamma)$$

$$E[\varepsilon_q^2] = \sigma^2_{\varepsilon_q} = \sum_{j=1}^{k} \sigma^2_{\varepsilon_{q_j}}$$

Mean Square Error

- Total error
  $\varepsilon = \varepsilon_d + \varepsilon_q$
- MSE
  $E[\varepsilon^2]=E[\varepsilon_d^2]+E[\varepsilon_q^2]$

Relative Error Model

- In the error model, the displacement and quantization are measured in absolute error
- Those errors are not related to the size of the object

$\varepsilon = 2$, $L = 10$

Absolute error = 2

Relative error = 20%

$\varepsilon = 2$, $L = 15$

Absolute error = 2

Relative error = 13.3%

Relative Error Modeling

- Displacement Error

$\varepsilon_{dx} = (\varepsilon_{du1} - \varepsilon_{du2})$

$\varepsilon_{dy} = (\varepsilon_{dv1} - \varepsilon_{dv2})$

$\varepsilon_d \approx (\varepsilon_{dx}\cos(\gamma) + \varepsilon_{dy}\sin(\gamma))$

- Total relative displacement error for k lines is:

$$\varepsilon_d = \sum_{j=1}^{k} \varepsilon_{d_j} \Big/ \sum_{j=1}^{k} L_j$$

$$E[\varepsilon_d^2] = (\sigma_{\varepsilon_d}^2 + \eta_{\varepsilon_d}^2) \Big/ \left(\sum_{j=1}^{k} L_j\right)^2$$

Relative Error Modeling Cont.

- Quantization Error

$$\varepsilon_q = (L_q - L) = \left(\sqrt{(L\cos\gamma + \varepsilon_{qx})^2 + (L\sin\gamma + \varepsilon_{qy})^2}\right) - L$$

$\varepsilon_{qx} = (L_{qx} - L_x)$

$\varepsilon_{qy} = (L_{qy} - L_y)$

$\varepsilon_q \approx \varepsilon_{qx}\cos(\gamma) + \varepsilon_{qy}\sin(\gamma)$

$$E[\varepsilon_q^2] = \sigma_{\varepsilon q}^2 \approx (1/6)(r_x^2\cos^2\gamma + r_y^2\sin^2\gamma)$$

Relative Error Modeling Cont.

- Total relative quantization error for k lines is:

$$\varepsilon_q = \sum_{j=1}^{k} \varepsilon_{q_j} \Big/ \sum_{j=1}^{k} L_j$$

$$E[\varepsilon_q^2] = \sigma_{\varepsilon_q}^2 \Big/ \left(\sum_{j=1}^{k} L_j\right)^2$$

Relative Mean Square Error (RMSE)

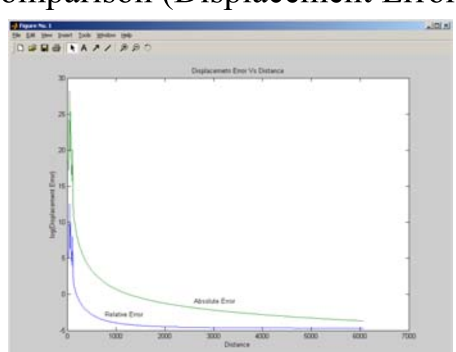
- Total error

$\varepsilon = \varepsilon_d + \varepsilon_q$

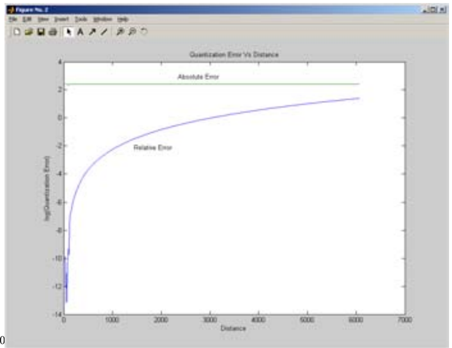
- RMSE

$$E[\varepsilon^2] = E[\varepsilon_d^2] + E[\varepsilon_q^2] = (\sigma_{\varepsilon_d}^2 + \eta_{\varepsilon_d}^2) \Big/ \left(\sum_{j=1}^{k} L_j\right)^2 + \sigma_{\varepsilon_q}^2 \Big/ \left(\sum_{j=1}^{k} L_j\right)^2$$

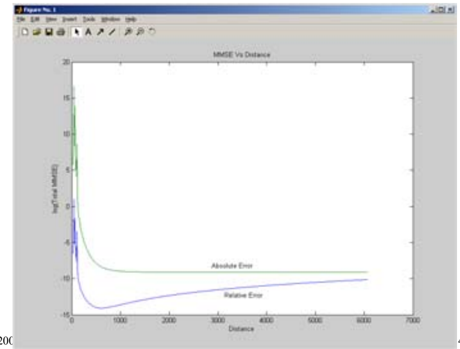
Comparison (Displacement Error)



Comparison (Quantization Error)



Comparison (MSE)



Error Analysis of Stereo System

- The RMSE of the total error ($E[\varepsilon^2]$) of one image is formulated as following

$$E[\varepsilon_d] = \eta_{\varepsilon_d}$$
$$Var[\varepsilon_d] = \sigma^2_{\varepsilon_d}$$
$$E[\varepsilon_q] = \eta_{\varepsilon_q}$$
$$Var[\varepsilon_q] = \sigma^2_{\varepsilon_q}$$

$$\begin{aligned} Cov(\varepsilon_d, \varepsilon_q) &= E[\varepsilon_d \varepsilon_q] - E[\varepsilon_d]E[\varepsilon_q] \\ &= E[\varepsilon_d]E[\varepsilon_q] - E[\varepsilon_d]E[\varepsilon_q] \\ &= 0 \end{aligned}$$

$$E[\varepsilon_d + \varepsilon_q] = \eta_{\varepsilon_d} + \eta_{\varepsilon_q}$$

$$\begin{aligned} Var[\varepsilon_d + \varepsilon_q] &= Var[\varepsilon_d] + Var[\varepsilon_q] + 2Cov(\varepsilon_d, \varepsilon_q) \\ &= \sigma^2_{\varepsilon_d} + \sigma^2_{\varepsilon_q} \end{aligned}$$

$$\begin{aligned} E[\varepsilon^2] &= Var[\varepsilon] + E^2[\varepsilon] \\ &= Var[\varepsilon_d + \varepsilon_q] + E^2[\varepsilon_d + \varepsilon_q] \\ &= \sigma^2_{\varepsilon_d} + \sigma^2_{\varepsilon_q} + (\eta_{\varepsilon_d} + \eta_{\varepsilon_q})^2 \\ &= \sigma^2_{\varepsilon_d} + \eta^2_{\varepsilon_d} + \sigma^2_{\varepsilon_q} + \eta^2_{\varepsilon_q} + 2\eta_{\varepsilon_d}\eta_{\varepsilon_q} \\ &= E[\varepsilon_d^2] + E[\varepsilon_q^2] + 2\eta_{\varepsilon_d}\eta_{\varepsilon_q} \end{aligned}$$

Error Analysis of Stereo System Cont.

- Let $\boldsymbol{\varepsilon_L}$ and $\boldsymbol{\varepsilon_R}$ be the random variables of the left and right images respectively

$$\begin{aligned} E[\varepsilon_L] &= E[\varepsilon_{L_d} + \varepsilon_{L_q}] \\ &= \eta_{\varepsilon_L} \end{aligned}$$
$$\begin{aligned} Var[\varepsilon_L] &= Var[\varepsilon_{L_d} + \varepsilon_{L_q}] \\ &= \sigma^2_{\varepsilon_L} \end{aligned}$$
$$\begin{aligned} E[\varepsilon_R] &= E[\varepsilon_{R_d} + \varepsilon_{R_q}] \\ &= \eta_{\varepsilon_R} \end{aligned}$$
$$\begin{aligned} Var[\varepsilon_R] &= Var[\varepsilon_{R_d} + \varepsilon_{R_q}] \\ &= \sigma^2_{\varepsilon_R} \end{aligned}$$

Correlatedness Stereo Error Model

- Let r be the correlation coefficient of RVs $\boldsymbol{\varepsilon_L}$ and $\boldsymbol{\varepsilon_R}$ is by definition the ratio
- The RMSE of the total error ($E[\varepsilon_S^2]$) of the stereo images is formulated as following

$$r = Cov(\varepsilon_L, \varepsilon_R) / \sigma_{\varepsilon_L}\sigma_{\varepsilon_R}$$

$$Cov(\varepsilon_L, \varepsilon_R) = E[\varepsilon_L \varepsilon_R] - E[\varepsilon_L]E[\varepsilon_R]$$
$$|Cov(\varepsilon_L, \varepsilon_R)| \le \sigma_{\varepsilon_L}\sigma_{\varepsilon_R}$$
$$|r| \le 1$$

$$\varepsilon_S = \varepsilon_L + \varepsilon_R$$

$$\begin{aligned} E[\varepsilon_S] &= E[\varepsilon_L + \varepsilon_R] \\ &= E[\varepsilon_L] + E[\varepsilon_R] \\ &= \eta_{\varepsilon_L} + \eta_{\varepsilon_R} \end{aligned}$$

$$\begin{aligned} Var[\varepsilon_S] &= Var[\varepsilon_L + \varepsilon_R] \\ &= Var[\varepsilon_L] + Var[\varepsilon_R] + 2Cov(\varepsilon_L, \varepsilon_R) \\ &= \sigma^2_{\varepsilon_L} + \sigma^2_{\varepsilon_R} + 2r\sigma_{\varepsilon_L}\sigma_{\varepsilon_R} \end{aligned}$$

Correlatedness Stereo Error Model Cont.

- Assume r ∝ 1/δ, where δ is the baseline (distance between the centers of projecting of the two cameras)

$$r = k\frac{1}{\delta}$$

- where *k* is an arbitrary constant and k ≤ δ

$$\begin{aligned} E[\varepsilon_S^2] &= Var[\varepsilon_S] + E^2[\varepsilon_S] \\ &= Var[\varepsilon_L + \varepsilon_R] + E^2[\varepsilon_L + \varepsilon_R] \\ &= \sigma^2_{\varepsilon_L} + \sigma^2_{\varepsilon_R} + 2r\sigma_{\varepsilon_L}\sigma_{\varepsilon_R} + (\eta_{\varepsilon_L} + \eta_{\varepsilon_R})^2 \\ &= \sigma^2_{\varepsilon_L} + \sigma^2_{\varepsilon_R} + \eta^2_{\varepsilon_L} + \eta^2_{\varepsilon_R} + 2r\sigma_{\varepsilon_L}\sigma_{\varepsilon_R} + 2\eta_{\varepsilon_L}\eta_{\varepsilon_R} \\ &= \sigma^2_{\varepsilon_L} + \sigma^2_{\varepsilon_R} + \eta^2_{\varepsilon_L} + \eta^2_{\varepsilon_R} + 2(r\sigma_{\varepsilon_L}\sigma_{\varepsilon_R} + \eta_{\varepsilon_L}\eta_{\varepsilon_R}) \\ &= E[\varepsilon_L^2] + E[\varepsilon_R^2] + 2(r\sigma_{\varepsilon_L}\sigma_{\varepsilon_R} + \eta_{\varepsilon_L}\eta_{\varepsilon_R}) \end{aligned}$$

Uncorrelatedness Stereo Error Model

- Assume the two RVs ε_L and ε_R are uncorrelated. Hence ,

$$\mathrm{Cov}(\varepsilon_L, \varepsilon_R) = 0$$
$$r = 0$$
$$\mathrm{E}[\varepsilon_L \varepsilon_R] = \mathrm{E}[\varepsilon_L]\mathrm{E}[\varepsilon_R]$$

- The RMSE of the total error ($\mathrm{E}[\varepsilon_S^2]$) of the stereo images is formulated as following

$$\begin{aligned}\mathrm{E}[\varepsilon_S^2] &= \mathrm{Var}[\varepsilon_S] + \mathrm{E}^2[\varepsilon_S] \\ &= \mathrm{Var}[\varepsilon_L + \varepsilon_R] + \mathrm{E}^2[\varepsilon_L + \varepsilon_R] \\ &= \sigma_{\varepsilon_L}^2 + \sigma_{\varepsilon_R}^2 + (\eta_{\varepsilon_L} + \eta_{\varepsilon_R})^2 \\ &= \mathrm{E}[\varepsilon_L^2] + \mathrm{E}[\varepsilon_R^2] + 2\eta_{\varepsilon_L}\eta_{\varepsilon_R}\end{aligned}$$

Visual Constraints

- To limit the positions and orientations of camera, visual constraints are introduced.
- The visual constraints are formulated as:
  - Resolution Constraint
  - Focus Constraint
  - Field-of-View Constraint
  - View-of-Angle Constraint
  - Visibility Constraint

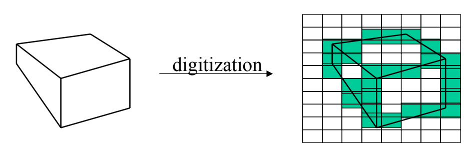
Resolution Constraint

- To make sure the entities can be inspected accurately

$$g_1 : \frac{w}{dl}[(\vec{r}_O - \vec{r}_A)\cdot\vec{v}][(\vec{r}_O - \vec{r}_B)\cdot\vec{v}] - \left\|\vec{v}\times[\vec{e}\times(\vec{r}_O - \vec{r}_A)]\right\| \le 0$$

$\vec{r}_o$: position vector of the front nodal point of the lens
$\vec{r}_A$: position vectors of the line endpoints
$\vec{r}_B$: position vectors of the line endpoints
$\vec{e}$: unit vector along the line segment
$\vec{v}$: unit vector along the optical axis in viewing direction
l : the length of minimum entity to be resovled
w : the minimum required lenth of the entity projected on teh image
d : distance from the back nodal point of the lens to the image plane

Resolution Constraint Cont.



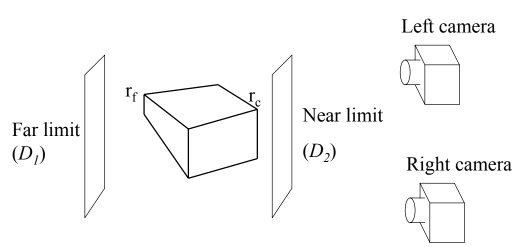
Focus Constraint

- To keep the desire inspected features in focus

$$D_1 = \frac{afd}{a(d-f)-cf} \qquad g_{2a} : D_2 - (\vec{r}_C - \vec{r}_O)\cdot\vec{v} \le 0$$
$$D_2 = \frac{afd}{a(d-f)+cf} \qquad g_{2b} : (\vec{r}_f - \vec{r}_O)\cdot\vec{v} - D_1 \le 0$$

$\vec{r}_f$: position vector of the farthest feature vertex from the front nodal point of the lens along the viewing direction
$\vec{r}_c$: position vector of the closest feature vertex from the front nodal point of the lens along the viewing direction
f : focal length
c : radius of the blur circle
D_1 : farthest distance of depth of field
D_2 : nearst distance of depth of field

Focus Constraint Cont.



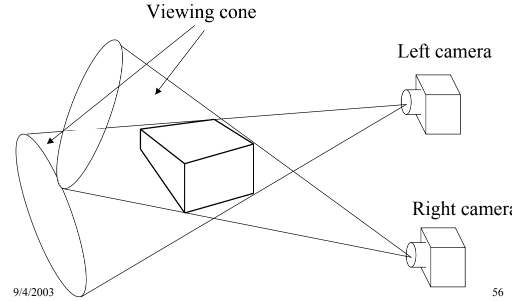
Field-of-View Constraint

- To make sure all the desire entities are projected inside the image plane

$$\alpha = 2\tan^{-1}\left(\frac{I_{\min}}{2d}\right) \qquad g_3 : \left\|\vec{r}_m - \vec{r}_o\right\| \cos\left(\frac{\alpha}{2}\right) - (\vec{r}_m - \vec{r}_o)\cdot\vec{v} \le 0$$

α : field - of - view angle
$I_{\min}$: minimum dimension of the image plane
$\vec{r}_m$: position vector of the extreme vertex, extreme vertex is a vertex that has the largest angle between the viewing direction vector and the vector $(\vec{r}_m - \vec{r}_o)$

Field-Of-View Constraint Cont.



View-of-Angle Constraint

- Object features cannot be measured when viewed from directions approaching coplanar
- To make sure the entities projected in the image can be distingue and measured

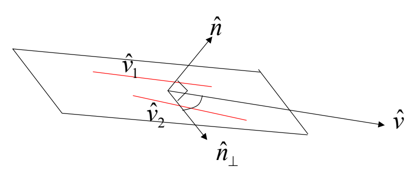
$$g_4 : \delta - \sin^{-1}(\vec{n}\cdot\vec{v}) \le 0$$

δ : minimum angle between the viewing direction and the plane formed by two desired entities
$\vec{n}$: normal vector of the plane formed by the desired entities

View-of-Angle Constraint Cont.

$$\hat{n} = \hat{v}_1 \times \hat{v}_2$$
$$\sin^{-1}(\hat{n}\cdot\hat{v}) >= \delta_{inc}$$

Where δ_{inc} is the minimum angle between the plane and the viewing direction



Visibility Constraint

- To make sure all the desired features can be seen and are not occluded

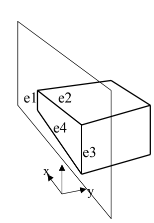
$$g_5 : \vec{r}_o \cdot P_i \ge 0$$

P_i : plane equations to form the viewing volume

Visibility Constraint Cont.

Example:

To see entities e1, e2, e3, e4, the camera must satisfy equation y < 0



Nonlinear Program

- To determine the optimal camera pose, the approach is to use nonlinear programming.
- The nonlinear program is formulated with visual constraints
  - Resolution
  - field-of-view
  - Focus
  - view-of-angle
  - visibility
- objective function
  - RMSE of the desired entities

Nonlinear Program Cont.

- Optimal camera pose
  - minimal effects of quantization and displacement errors on dimensioning of linear geometric entities.
  - The quantization and displacement errors are formulated as mean squared error.
- minimize $E[\varepsilon^2] = f(t_x, t_y, t_z, \Phi, \theta, \Psi)$

  subject to:
  - g1j <= 0 (pixel resolution), for j=1 to k
  - g2a <= 0 (focus)
  - g2b <= 0
  - g3 <= (field of view)
  - g4j <= 0 (incidence angle constraints), for j=1 to ${}_kC_2(\frac{k!}{2!(k-2)!})$
  - g5i <= 0 (visibility), for i=1 to m

Nonlinear Program Cont.

minimize $E[\varepsilon^2] = f(t_{xL}, t_{yL}, t_{zL}, \phi_L, \theta_L, \psi_L, t_{xR}, t_{yR}, t_{zR}, \phi_R, \theta_R, \psi_R)$

subject to

$g_{L1j} \le 0$ (resolution) for j = 1 to k	$g_{R1j} \le 0$ (resolution) for j = 1 to k
$g_{L2a} \le 0$ (focus)	$g_{R2a} \le 0$ (focus)
$g_{L2b} \le 0$ (focus)	$g_{R2b} \le 0$ (focus)
$g_{L3} \le 0$ (field - of - view)	$g_{R3} \le 0$ (field - of - view)
$g_{L4i} \le 0$ (view - of - angle) for i = 1 to ${}_kC_2$	$g_{R4i} \le 0$ (view - of - angle) for i = 1 to ${}_kC_2$
$g_{L5i} \le 0$ (visibility) for i = 1 to m	$g_{R5i} \le 0$ (visibility) for i = 1 to m

Initial Feasible camera pose

- It is no guarantee that the nonlinear program will determine a solution.
- And the final solutions depend on the initial camera poses

NLP Example

- Ex1
  - Initial pose
    - camera position/orientation: 1216.193661 -954.187115 -1220.729648 0.513804 -0.456819 -0.811894
  - Optimal pose
    - camera position/orientation: 282.725838 -499.690340 -1469.377994 0.927833 0.151262 -0.342271
    - Optimal Value: 0.242909
- Ex2
  - Initial pose
    - camera position/orientation: 499.146112 -680.528149 -1361.056298 0.244979 -0.237941 -0.510331
  - Optimal pose
    - camera position/orientation: 492.216499 -666.899532 -1368.051894 -1.212159 -0.525523 0.145824
    - Optimal Value: 0.245218
- Ex3
  - Initial pose
    - camera position/orientation: 70.213576 -960.212503 -2105.580459 0.026566 -0.026556 -0.428443
  - Optimal pose
    - camera position/orientation: 470.466854 -641.647819 -1384.297776 -1.134582 -0.511611 0.101482
    - Optimal Value: 0.242944

Initial Feasible Camera Pose Determination

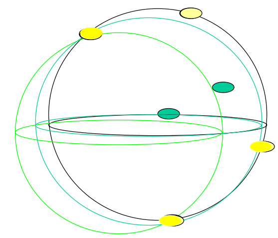
- Outline to determine initial feasible camera pose
  1. Construct the minimum bounding sphere
  2. Determine the minimum distance that satisfies focus and field of view constraints
  3. Determine the viewing direction
  4. Find camera pose that is closest to the viewing direction

Minimum bounding sphere

- Algorithm(modification of Elzinga-Hearn Algorithm)
  - Input: A set P of 3D points
  - Output: The center and radius of a sphere that covers the points in P, and points Pi, Pj, Pk, and Pl that are on the sphere

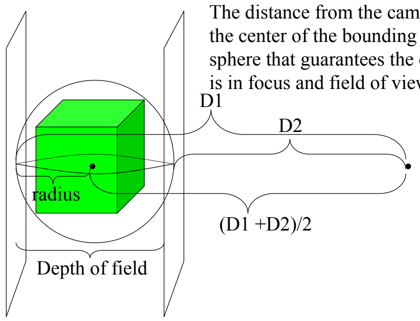
1. Choose two points, P_i and P_j that are farthest apart
2. Construct the sphere whose center $C_2((P_i + P_j)/2)$
   1. If the sphere contain all points, then the center of the sphere is C_2
   2. Else, find a point P_k is the farthest away from C_2
3. Use the three points (P_i ,P_j, P_k) to determine the center C_3
   1. If the sphere contain all points, then the center of the sphere is C_3
   2. Else, find a point P_l is the farthest away from C_3
4. Use the there points (P_i ,P_k, P_l) to determine the center C_{3a}
   1. If the sphere contain all points, then the center of the sphere is C_{3a}
   2. Else, use the there points (P_j ,P_k, P_l) to determine the center C_{3b}
      1. If the sphere contain all points, then the center of the sphere is C_{3b}
      2. Else, use the four points (P_i , P_j ,P_k, P_l) to construct the sphere whose center is C_4

Minimum Bounding Sphere Example



Determine Minimum Distance

The distance from the camera to the center of the bounding sphere that guarantees the object is in focus and field of view



Minimum focus distance

$$D_1 = \frac{Daf}{af - c(D-f)}$$

$$D_2 = \frac{Daf}{af + c(D-f)}$$

$$Depth\ \ of\ \ field(DOF) = D_1 - D_2 = 2r$$

Where:

D : focus distance
a : lens's aperture
f : focal length
c : blur circle radius
r : radius of minimum bounding sphere

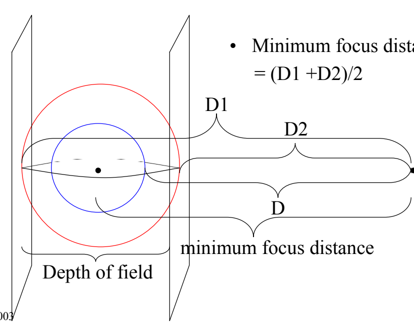
$$2r = Daf\left(\frac{1}{af - c(D-f)} - \frac{1}{af + c(D-f)}\right)$$

$$D^2(2afc + 2rc^2) + D(-2af^2c - 4rfc^2) + 2rf^2(c^2 - a^2) = 0$$

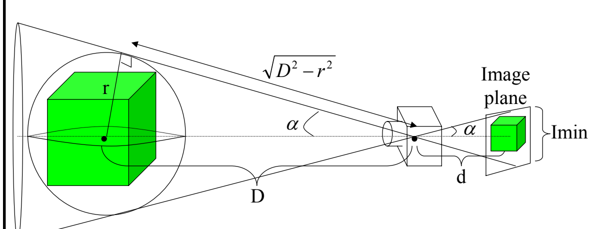
$$D = \frac{2acf^2 + 4rfc^2 + \sqrt{(-2acf^2 - 4rfc^2)^2 - 16rcf^2(af + rc)(c^2 - a^2)}}{4afc + 4rc^2}$$

Minimum focus distance Cont.

- Minimum focus distance = (D1 +D2)/2



Minimum field of view distance



$$\tan\alpha = \frac{r}{\sqrt{D^2 - r^2}}$$

$$\tan\alpha = \frac{I\min}{2d}$$

Where:

d : effective focal length
α : field of view angle

Minimum field of view distance Cont.

$$\frac{r}{\sqrt{D^2 - r^2}} = \frac{I\min}{2d}$$

$$\frac{1}{d} + \frac{1}{D} = \frac{1}{f}$$

$$\frac{r}{\sqrt{D^2 - r^2}} = \frac{I\min(D-f)}{2fD}$$

$$(D^2 - r^2)(D-f)^2 I\min^2 = 4f^2D^2r^2$$

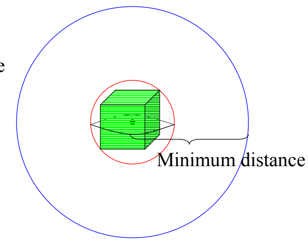
$$D^4 - D^3 2f - D^2(r^2 + \frac{4f^2r^2}{I\min^2} - f^2) + D2fr^2 - f^2r^2 = 0$$

Minimum field of view distance = D

Minimum Distance

Minimum distance = max {minimum focus distance, minimum field of view distance}

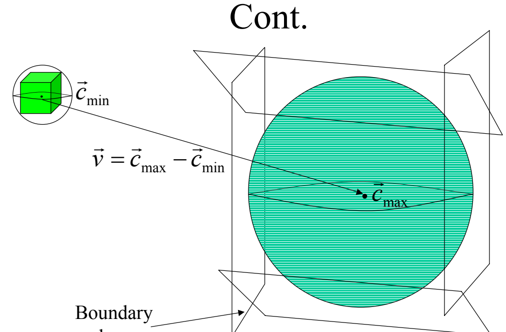
Any camera position outside the blue circle, satisfies the field of view and focus constraints



Determine Viewing Direction

- Construct a maximum sphere that is bounded by the visibility planes
- The vector from the maximum sphere center and the minimum sphere center is the viewing direction

Determine Viewing Direction Cont.



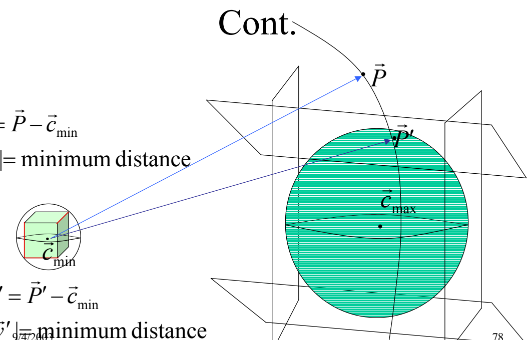
Determine feasible camera pose

- The intersection of the minimum distance and the viewing direction gives you a feasible camera pose.
  - assume this is the best pose.
- Sometimes this pose is outside the visibility volume.
- Need a way to find a new pose as close as possible to the original pose that is inside the visibility volume

Determine feasible camera pose Cont.

$\vec{v} = \vec{P} - \vec{c}_{\min}$

$|\vec{v}|$= minimum distance



$\vec{v}' = \vec{P}' - \vec{c}_{\min}$

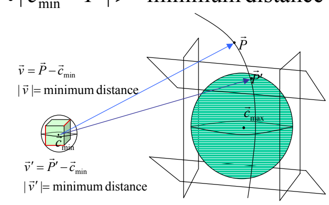
$|\vec{v}'|$= minimum distance

Determine feasible camera pose Cont.

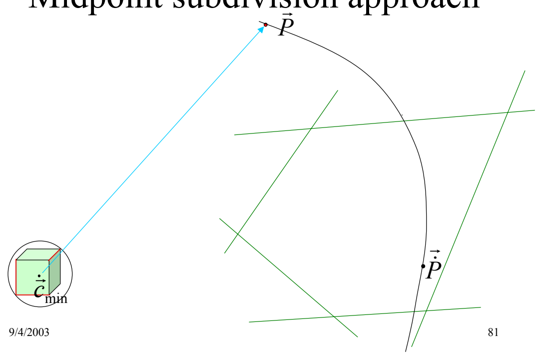
- Find camera pose ($\vec{P}'$) is most close the viewing direction
- Two approaches:
  - Nonlinear program
  - Midpoint subdivision

Nonlinear program approach

- minimize $|\vec{P} - \vec{P}'| = f(\vec{P}, \vec{P}')$
  subject to:
  - $P_i \cdot \vec{P}' >= 0$ Where $P_i(x,y,z) = A_i x + B_i y + C_i z + D_i = 0$
  - $|\vec{c}_{\min} - \vec{P}'| >=$ minimum distance

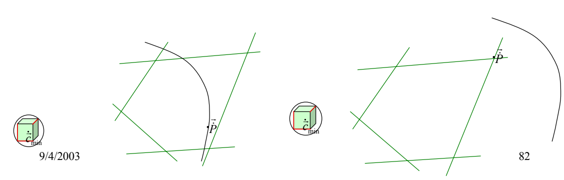


Midpoint subdivision approach

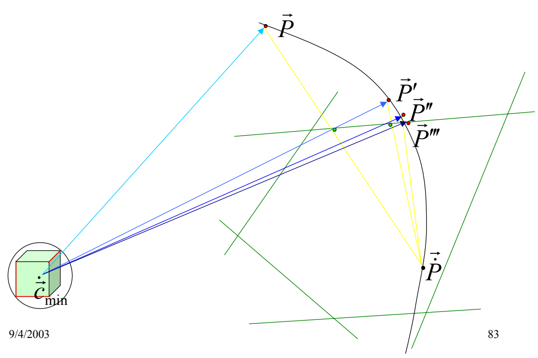


Midpoint subdivision approach cont.

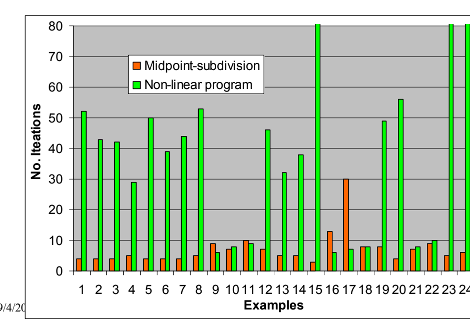
- maximize $|\dot{\vec{P}} - \vec{c}_{\min}| = f(\dot{\vec{P}}, \vec{c}_{\min})$
  subject to:
  - $P_i \cdot \dot{\vec{P}} >= 0$ Where $P_i(x,y,z) = A_i x + B_i y + C_i z + D_i = 0$
  - minimum distance - $|\vec{c}_{\min} - \dot{\vec{P}}| >= 0$



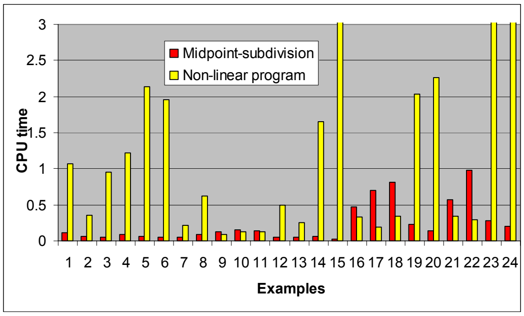
Midpoint subdivision approach Cont.



Comparison between the two approaches

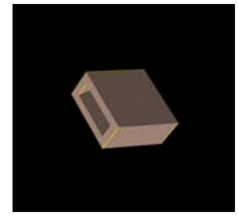


Comparison between the two approaches cont.



Initial feasible camera pose Example

NODE: N0057 (E0000 E0005 E0008 E0013)[1 4 5 17 20]
camera position/orientation: 1229.699622 -1234.295868 -1693.115135 0.397682 -0.369495 -0.726734
viewing direction: -0.5062 0.5081 0.6969
Optimal Value: 0.909082
fov angle: 11.6934
Dmax: 2.4584e+003
effective focal length: 25.2669
aperture: 25
depth of field: 2458.384081 2280.849262

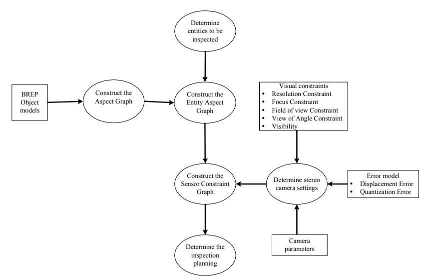


Initial Feasible Camera Pose

Automatic Visual Inspection Planning System

- By integration
  - aspect graph (AG)
  - entity-based aspect graph (EAG)
  - sensor constraint graph (SCG)
  - visual constraints
  - relative error model
  - Stereo cameras

System Architecture



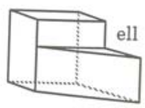
Aspect Graph

- In order to determine the regions for camera placement and the geometric entities are seen within that viewing volume.
- The aspect graph of the object is generated.
- From the aspect graph, each node has the information of the geometric entities can be seen by the viewing volume assigns in that node.
- The viewing volume is the intersection of the hyperplanes.

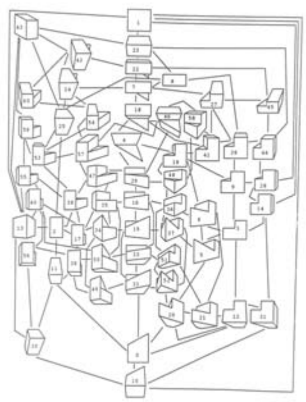
Aspect Graph Cont.

- Nodes and arcs (C, A)
- C is a set of the characteristic view domain
- A is a set of the adjacent pairs of nodes which link the nodes together
- Two nodes are adjacent to each other when they have an arc between them.
- Physically, it means that two viewing volumes touch each and they share at least one hyperplane

AG Example



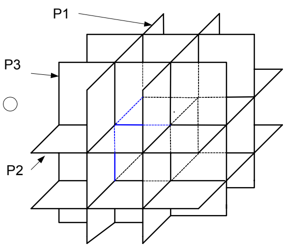
- Aspect graph of ell



Aspect Graph (Stewman)

- Viewing volumes as the intersection of hyperplanes

Ex:



Viewing Volume:

H1H2H3

Entity Aspect Graph

- In aspect graph, it represents all the characteristic views of an object.
- However, only small portions of characteristic views are necessary.
- Since not all the geometric entities of the object are interested, only several entities are required to be inspected.
- In order to reduce computational cost and complexity, the aspect graph is simplified and it contracts to entity aspect graph.

Entity Aspect Graph (Yang)

- Viewing regions for a subset of entities on the object
- EAG simplifies the Aspect Graph
- In the EAG, it has four elements (E, V, O, A).
  - E is a set of entity of interest.
  - V is a set of viewing domains.
  - O is a set of lists of observable entities.
  - A is a set of adjacent pairs of entity viewing domain

Entity Aspect Graph (Yang) Cont.

- Operations on EAG:
  - Construction
  - Contraction

Converting AG to EAG (idea)

- Assume initial AG is an EAG that contains all entities
- Apply contraction algorithm to AG to create desired EAG

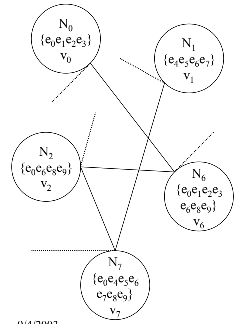
Converting AG to EAG (Cont.)

- Algorithm for contraction of aspect graph $AG^1(E^1, V^1, O^1, A^1)$ to construct $EAG^2(E^2, V^2, O^2, A^2)$ with $E^2 \subseteq E^1$

Input: $AG^1(E^1, V^1, O^1, A^1), E^2$
Output: $EAG^2(E^2, V^2, O^2, A^2)$

1. **Initialize EAG^2*
   Set $V^1 = V^2, O^1 = O^2, A^1 = A^2$
2. **Determine the visibility of the entities in E^2 for each element in V^2*
   For every element of O^2
   update the list of observable entities (O_{vi}^2) to be the intersection of the original list of observable entities (O_{vi}^1) and E^2. ($O_{vi}^2 = O_{vi}^1 \cap E^2$)
3. **Contract the adjacent viewing domains if their observable entities are the same*
   For every element of $A^2, (V_i^2, V_j^2)$
   If two adjacent viewing domains (V_i^2, V_j^2) of their observable entities are the same ($O_{vi}^2 = O_{vj}^2$).
   Then contracts V_i^2 and V_j^2 by replace them with a new viewing domain, $V_k^2 = V_i^2 \cup V_j^2$.
   Set the corresponding list of observable entities (O_{vk}^2) with O_{vi}^2.
   Remove O_{vi}^2 and O_{vj}^2 from O^2.
   Remove the current arc of EAG^2, (V_i^2, V_j^2), from A^2.
   For every element of A^2, substitute V_i^2 and V_j^2 by V_k^2.

Converting AG to EAG Example:

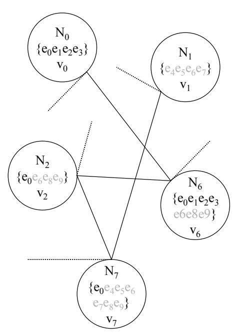


$\cap$ EOI = $\{e_0e_1e_2e_3\}$

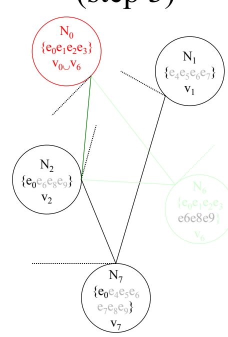
1) Intersect EOIs with entities in AG

Converting AG to EAG Example:

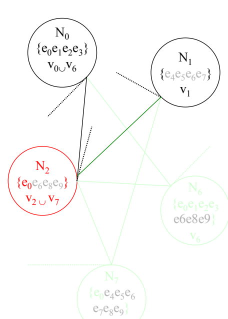
2) merge nodes with same observable entities



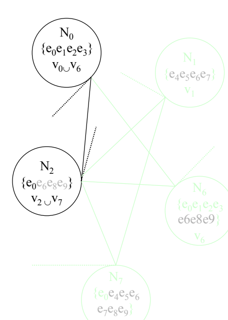
Converting AG to EAG Example: (step 3)



Converting AG to EAG Example:

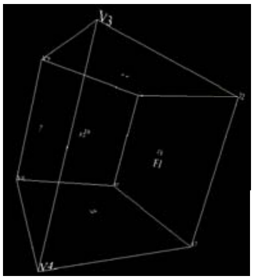


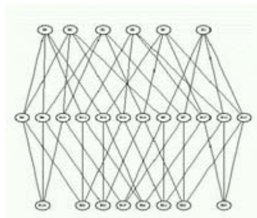
Converting AG to EAG Example:



Converting AG to EAG Cont.

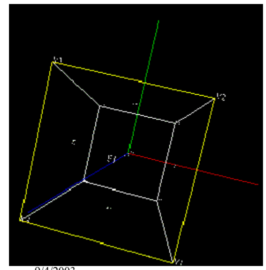
- Convert AG to EAG Example (Cube)
- EOI = {e0,e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11}

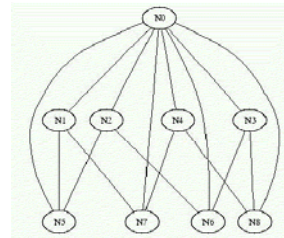




Converting AG to EAG Cont.

- EOI = {e0,e1, e2, e3}





Merging viewing regions

- How are two nodes merged?
- How do you calculate the union of two viewing regions?
  - Viewing regions must form convex volumes in order to be formulated as linear constraints in the NLP formulation
  - It is possible for the valid viewing regions to form concave volumes

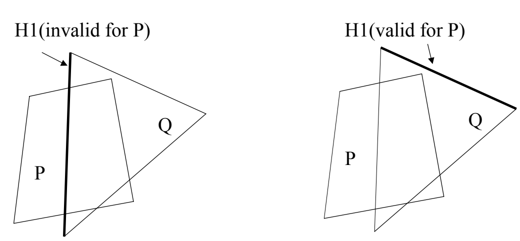
Determining convexity of union of viewing regions

- The viewing regions (Convex H-Polyhedra) is the intersection of the hyperplanes.
- The viewing regions is represented as following
- **Sp2Sp6Sp7Sp4Hp9**
  - p2: $y - 0.25 = 0$
  - p6: $-x - 1 = 0$
  - p7: $x - 1 = 0$
  - p4: $-y - 1 = 0$
  - p9: $0.7071y + 0.7071z - 0.8838 = 0$
- Where H/S represented the positive or the negative normal of the hyperplanes respectively.
  - p2: $y - 0.25 < 0$
  - p6: $-x - 1 < 0$
  - p7: $x - 1 < 0$
  - p4: $-y - 1 < 0$
  - p9: $0.7071y + 0.7071z - 0.8838 > 0$

Determining convexity of union of viewing regions Cont.

- Let P be a convex polyhedron in R^d.
- Then P can be represented as an inequality equation.
  - $a^Tx <= b$
- The inequality equation is valid for P if it is satisfied by all points in P
- Conversely, the inequality equation is invalid for P if there exist a point on the other side of the inequality that is in P

Example valid/invalid inequalities



Determining convexity of union of viewing regions Cont.

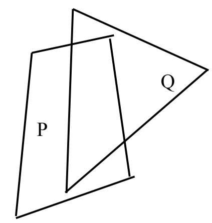
- Let P and Q be (possibly unbounded) H-Polyhedra

$$P=\{x\in\Re^d : Ax\le\alpha\},$$
$$Q=\{x\in\Re^d : Bx\le\beta\}.$$

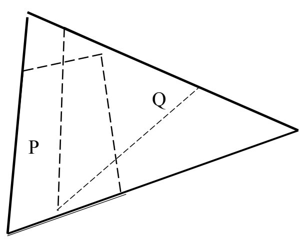
$$env(P,Q)=\{x\in\Re^d : \overline{A}x\le\overline{\alpha}, \overline{B}x\le\overline{\beta}\}$$

where $\overline{A}x \le \overline{\alpha}$ is the subsystem of $Ax \le \alpha$ that not valid for the other polyheron Q, $\overline{B}x \le \overline{\beta}$ defined similarly with respect to P

Example: env(P,Q) original

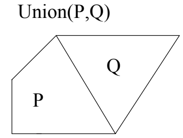


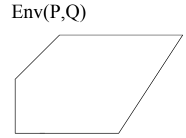
Example: env(P,Q)



Determining convexity of union of viewing regions Cont.

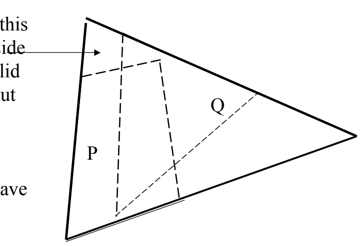
- Theorem:
  - Union(P,Q) is convex iff Union(P,Q)=env(P,Q)





Example: is union(P,Q) convex?

Any point in this region is outside a pair of invalid inequalities but inside the env(P,Q), therefore the union is concave



Determining convexity of union of viewing regions

1. Construct env(P,Q)

Let $\overline{A}x\le\overline{\alpha}$ $\overline{B}x\le\overline{\beta}$ be the set of removed constraints

Let $env(P,Q)=\{x:Cx\le\gamma\}$ be the resulting envelope

2. for each pair $\overline{A}_i x\le\overline{\alpha}_i$ $\overline{B}_j x\le\overline{\beta}_j$ do:

    E*= max (x)

    Subject To: $\overline{A}_i x\ge\overline{\alpha}_i+\varepsilon$, $\overline{B}_j x\ge\overline{\beta}_j+\varepsilon$, $Cx\le\gamma$

    If feasible return nonconvex

Endfor

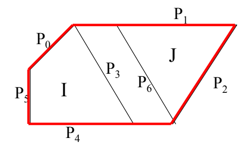
Return env(P,Q) // union(P,Q) is convex

Optimizing the union of regions

- Instead of using linear program to check the union of regions is convex or concave
- By checking the number of hyperplanes removed
- In AG, the viewing regions are disjoint and never overlap
- Union of regions in order to be convex
  - regions are adjacent to each other (share a common hyperplane)
  - number of hyperplanes removed = 1

Optimizing the union of regions Cont.

$I = \{P_0, P_3, P_4, P_5\}$
$J = \{P_1, P_2, P_6\}$



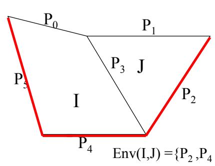
$Env(I,J) = \{P_0, P_1, P_2, P_4, P_5\}$
hyperplanes removed $= \{P_3, P_6\}$

Non adjacent
number of hyperplanes removed = 2

concave

Optimizing the union of regions Cont.

$I = \{P_0, P_3, P_4, P_5\}$
$J = \{P_1, P_2, P_3\}$



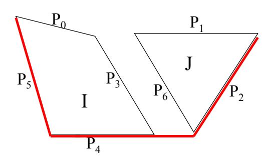
$Env(I,J) = \{P_2, P_4, P_5\}$
hyperplanes removed $= \{P_3, P_0, P_1\}$

adjacent
number of hyperplanes removed = 3

concave

Optimizing the union of regions Cont.

$I = \{P_0, P_3, P_4, P_5\}$
$J = \{P_1, P_2, P_6\}$



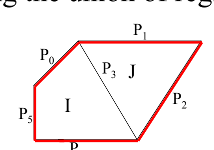
$Env(I,J) = \{P_2, P_4, P_5\}$
hyperplanes removed $= \{P_3, P_6, P_0, P_1\}$

Non adjacent
number of hyperplanes removed = 4

concave

Optimizing the union of regions Cont.

$I = \{P_0, P_3, P_4, P_5\}$
$J = \{P_1, P_2, P_3\}$



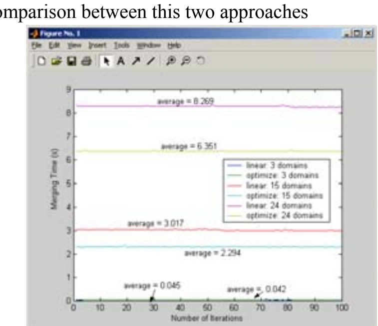
$Env(I,J) = \{P_0, P_1, P_2, P_4, P_5\}$
hyperplanes removed $= \{P_3\}$

adjacent
number of hyperplanes removed = 1

convex

Optimizing the union of regions Cont.

- Comparison between this two approaches

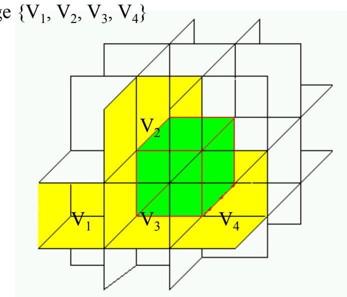


Merging viewing domains Algorithm

```
Input: G^i (A^i, V^i)
Output: G^o (A^o, V^o)
A: a set of the adjacent pairs of view domains represented as arcs.
V:  a set of viewing domains, {V_1, V_2, V_3, … }, in the three-dimensional space.
*Initialize
G^o = G^i
 i = 0
* merge the adjacent viewing domains if they are convex
while  i < size(V)
        i = i +1
        for every element of A (V_i, V_j)
                V_k = V_i ∪ V_j
                if V_k is convex  /* check if two viewing domains can merge */
                        arc(V_k) = arc(V_i) ∪ arc(V_j)
                          if i > j
                                    V_i = V_k
                                    V = V- V_j
                                    A = A ∪ arc(V_k)
                                    A = A- arc(V_j)
                                    i = i-1
                          else
                                    V_j = V_k
                                  V = V- V_i
                                    A = A ∪ arc(V_k)
                                    A = A- arc(V_j)
                          end
                          break   /*exit for loop*/
                end
        end
end
```

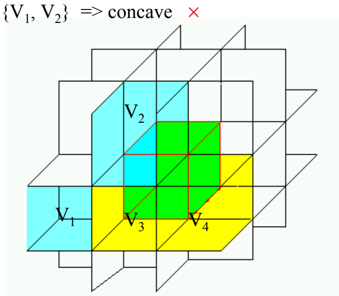

Viewing Domain Merging Example

- $L = \{V_1, V_2, V_3, V_4\}$
- Merge $\{V_1, V_2, V_3, V_4\}$



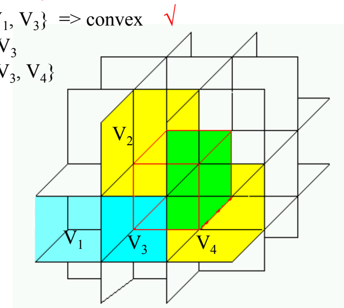
Viewing Domain Merging Example Cont.

- $L = \{V_1, V_2, V_3, V_4\}$
- Merge $\{V_1, V_2\}$ => concave ×



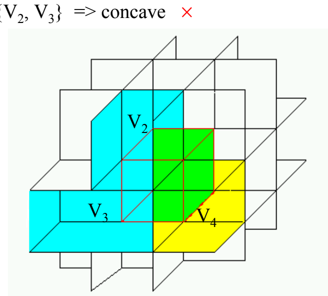
Viewing Domain Merging Example Cont.

- $L = \{V_1, V_2, V_3, V_4\}$
- Merge $\{V_1, V_3\}$ => convex √
- $V_3 = V_1 \cup V_3$
- $L = \{V_2, V_3, V_4\}$



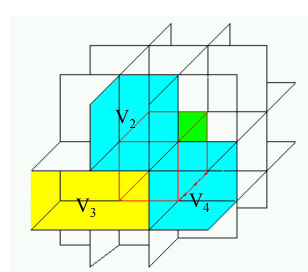
Viewing Domain Merging Example Cont.

- $L = \{V_2, V_3, V_4\}$
- Merge $\{V_2, V_3\}$ => concave ×



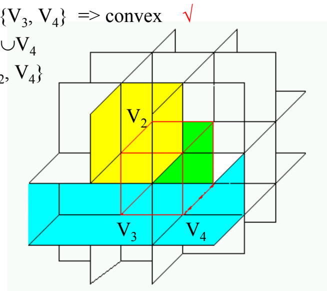
Viewing Domain Merging Example Cont.

- $L = \{V_2, V_3, V_4\}$
- Merge $\{V_2, V_4\}$ => concave ×



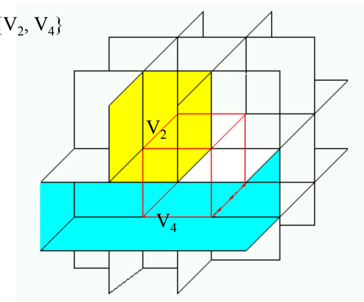
Viewing Domain Merging Example Cont.

- L = {V_2, V_3, V_4}
- Merge {V_3, V_4} => convex √
- $V_4 = V_3 \cup V_4$
- L = {V_2, V_4}

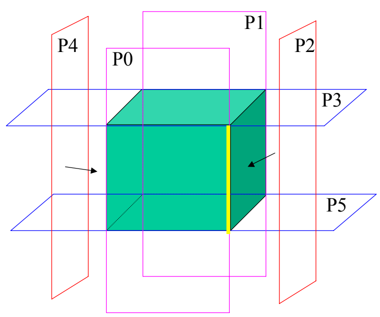


Viewing Domain Merging Example Cont.

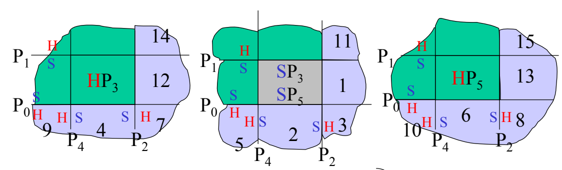
- L = {V_2, V_4}



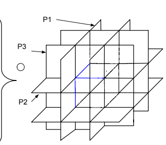
Viewing Domain Merging Example (Cube)



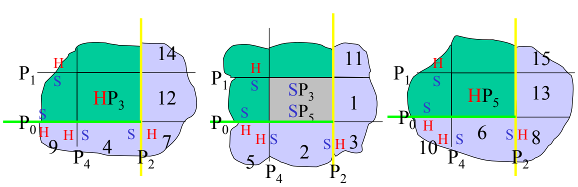
Viewing Domain Merging Example



1 (Sp0Hp2Sp3Sp1Sp5)	6 (Hp0Sp2Sp4Hp5)	11 (Hp2Sp3Hp1Sp5)
2 (Hp0Sp2Sp3Sp4Sp5)	7 (Hp0Hp2Hp3)	12 (Sp0Hp2Hp3Sp1)
3 (Hp0Hp2Sp3Sp5) ∪	8 (Hp0Hp2Hp5) ∪	13 (Sp0Hp2Sp1Hp5)
4 (Hp0Sp2Hp3Sp4)	9 (Hp0Hp3Hp4)	14 (Hp2Hp3Hp1)
5 (Hp0Sp3Hp4Sp5)	10 (Hp0Hp4Hp5)	15 (Hp2Hp1Hp5)



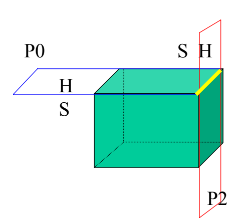
Viewing Domain Merging Example



1 (Sp0Hp2Sp3Sp1Sp5)	6 (Hp0Sp2Sp4Hp5)	11 (Hp2Sp3Hp1Sp5)
2 (Hp0Sp2Sp3Sp4Sp5)	7 (Hp0Hp2Hp3)	12 (Sp0Hp2Hp3Sp1)
3 (Hp0Hp2Sp3Sp5) ∪	8 (Hp0Hp2Hp5) ∪	13 (Sp0Hp2Sp1Hp5)
4 (Hp0Sp2Hp3Sp4)	9 (Hp0Hp3Hp4)	14 (Hp2Hp3Hp1)
5 (Hp0Sp3Hp4Sp5)	10 (Hp0Hp4Hp5)	15 (Hp2Hp1Hp5)

Merge → {1 (Hp2), 2 (Hp0Sp2)}

Viewing Domain Merging Example (Cube)



Sensor Constraint Graph

- Sensor constraint graph is generated by incorporating with the entity aspect graph, error model and visual constraints.
- Sensor Constraint Graph (SCG) is used to determine an inspection plan.
- SCG has two elements, which are nodes (N) and arcs (A).

Sensor Constraint Graph

- SCG node: 4 tuple (E,O,G,I)
  - E: set of desired geometric entities to be observed
  - O: objective function (RMSE)
  - G = { V, V’}
    - V visibility constraints
    - V’ focus, resolution, and field of view constraints
  - I: initial camera pose

Sensor Constraint Graph

- SCG Arcs
  - solid arcs
    - adjacent, yet disjoint visibility regions between two nodes
  - dashed arcs
    - overlapping visibility regions between two nodes

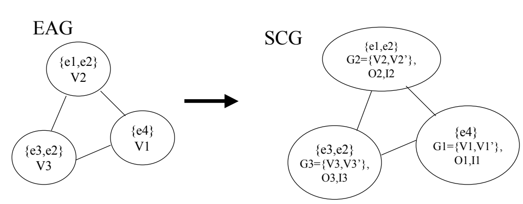
SCG operations

- SCG has a dynamic structure.
- To update the structure of SCG, two operation can be performed are expansion and contraction.
  - Expansion refers to “creation of subnodes”.
  - Contraction refers to “creation of supernodes”.

SCG construction

- Construct an EAG from the set of geometric entities
  - each node of EAG has set of visible desired entities and visibility constraints
- Define objective function for each node
- Define sensor constraints G for each node
- Choose initial camera pose I
- Link all nodes with solid arcs

Example: SCG construction

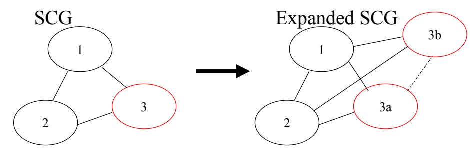


SCG expansion

- Creates subnodes
  - Subnodes' entity set is a subset of original node
  - RMSE function is defined in terms of such subset
  - Sensor constraints are also defined in terms of such subsets
  - Visibility constraints are the same as its original node

SCG expansion

- For multiple subnodes, the union of the desired entity set must be the same as the original nodes' entity set
- All subnodes are connected by dashed arcs

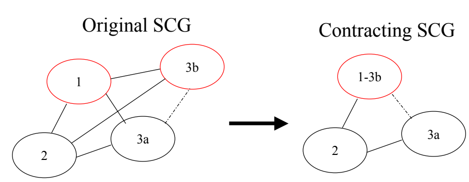


SCG contraction

- Creates supernodes
  - Two or more similar nodes are contracted
  - Supernode has the same entity set as original nodes
  - Same RMSE function
  - Same sensor constraints
  - Visibility constraints are the union of the visibility constraints of original nodes
  - Initial camera pose that results in Min(RMSE)

SCG contraction

- Neighboring nodes keep same relation with respect to supernode
- Example: contracting node1 and 3b



Subnode strategies

- Two types of failing optimizations
  - passing entities and failing entities in the set
  - only failing entities in the set
- Five strategies to handle these cases
  - Strategy 1: Pass/Fail
  - Strategy 2: One less
  - Strategy 3: Singleton
  - Strategy 4: Similar node with one less
  - Strategy 5: Similar node with singleton

Subnode strategies cont.

- Pass/Fail Strategy
  - create two subnodes, one with passing entities and one with failing entities
- One less Strategy
  - for failing optimization with k entities
  - create k subnodes containing k-1 entities each
- Singleton Strategy
  - create k subnodes each with 1 entity

Subnode strategies cont.

- Similar node with one-less strategy
  - Select all nodes neighboring N_o whose entity set is a subset of N_o's entity set
  - Create a subnode that matches the entity set of each neighboring node
  - If neighbors don't share any common entities, then use one less strategy
  - If the union of the neighbors' entity set doesn't include all the entities in N_o, then create an additional subnode with remaining entities

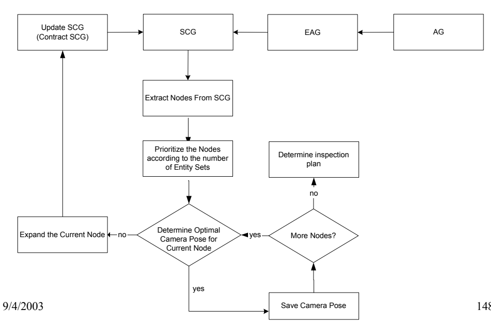
Subnode strategies cont.

- Similar nodes with singleton
  - same as before except using singleton strategy instead of one-less strategy

Visual Inspection Plan

- As the sensor constraint graph is generated, a visual inspection plan will be determined
  - minimize the number of camera poses
  - the smallest total relative mean square error

Visual Inspection Plan Cont.



Visual Inspection Plan Cont.

- Let
  - p be the number of camera poses generated from the plan
  - x_i = 1or 0 denote i^{th} camera pose
  - $S_i (F_i, E_i, O_i)$ is part of inspection plan for i=1 to p
    - F_i: optimal camera poses
    - E_i: the set of desired geometric entities
    - O_i: objective function (RMSE)
  - $a_{ij} = 1$ if entity E_j is in setting S_i
  - n be the total number of desired entities

Visual Inspection Plan Cont.

Case 1

$$Minimize \sum_{i=1}^{p} x_i$$

$$S.T.$$

$$Ax \geq 1$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{np} \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_p \end{bmatrix}$$

Case 2

$$Minimize \sum_{i=1}^{p} RMSE_i x_i$$

$$S.T.$$

$$Ax \geq 1$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{np} \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_p \end{bmatrix}$$

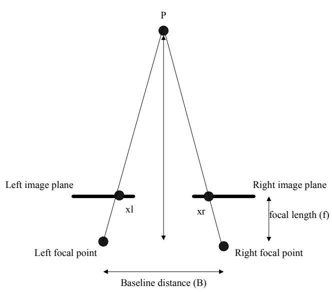
Visual Inspection Plan Cont.

- Visual inspection planning results
  - Rectangle object
  - 3-Pockets object
  - One-Hole object

3D Reconstruction

- To measure the length of the entities from the images, it requires a stereo camera system.
- After the entities are located, detected, and matched, they are mapped into 3D locations

Stereo Camera System



$$d = X_L - X_R$$

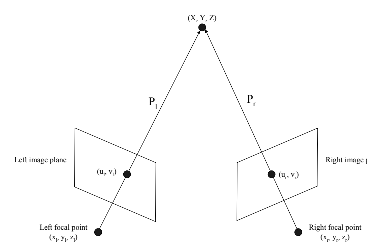
$$X = B\frac{X_L}{d}$$

$$Y = B\frac{Y_L}{d}$$

$$Z = f\frac{B}{d}$$

Where, d is the disparity, measures the difference in retinal position between the corresponding points in the two images
B is the baseline distance of the stereo system, distance between the centers of projecting of the two cameras

Reconstruction by Triangulation



$$\begin{bmatrix} u_l \\ v_l \end{bmatrix} = P_l \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\begin{bmatrix} u_r \\ v_r \end{bmatrix} = P_r \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Reconstruction by Triangulation Cont.

$$P_l = \begin{bmatrix} fm_x & 0 & x_0 \\ 0 & fm_x & y_0 \\ 0 & 0 & 1 \end{bmatrix} [R_l T_l] \qquad T_l = -R_l \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix}$$

$$P_r = \begin{bmatrix} fm_x & 0 & x_0 \\ 0 & fm_x & y_0 \\ 0 & 0 & 1 \end{bmatrix} [R_r T_r] \qquad T_r = -R_r \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix}$$

f is focal length,
m_x, m_y are the number of pixels per unit distance in the x and y direction respectively.
(x_0, y_0) is the coordinate of the optical axis in the image coordinate system.
R is the rotation matrix,
T is the translation matrix

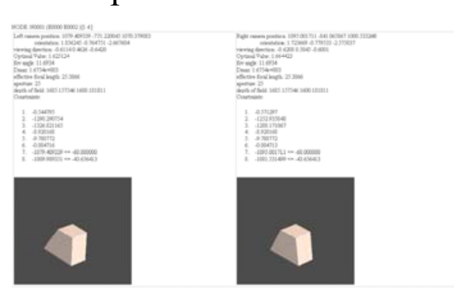
$$A\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0 \quad \text{where } A = \begin{bmatrix} u_l P_l^3 - P^1 \\ v_l P_l^3 - P^2 \\ u_r P_r^3 - P^1 \\ v_r P_r^3 - P^2 \end{bmatrix}_{4\times 4}$$

$$A'\begin{bmatrix} x \\ y \\ z \end{bmatrix} = b \quad \text{where } A = \left[A'_{4\times 3} \mid -b_{4\times 1} \right]$$

$$A' = [U]_{4\times 4} \begin{bmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_n \end{bmatrix}_{4\times 3} [V^T]_{3\times 3}$$

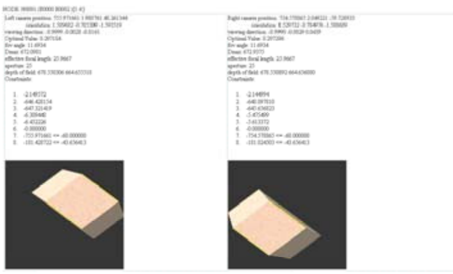
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = [V] \; [diag(1/w_j)] \; [U^T] \; [b]$$

Experimental Results



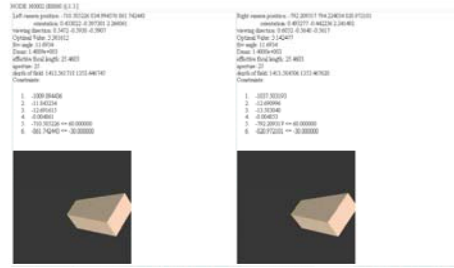
Initial Camera poses (E0, E2)

Experimental Results Cont.



Optimal Camera poses (E0, E2)

Experimental Results Cont.



Initial Camera poses (E0)

Experimental Results Cont.

Optimal Camera poses (E0)

Experimental Results Cont.

	Edges	Real Length (mm)	Measured Length (mm) (Initial Camera Poses)	Errors	Measured Length (mm) (Optimal Camera Poses)	Errors
Node 1	E0	60	45.9385	23.4359	60.0107	0.0178
	E2	60	43.9160	26.8067	60.6169	1.0282
Node 2	E0	60	52.3954	12.6743	59.3284	1.1193
Node 3	E2	60	44.7676	25.3873	59.7838	0.3603

Experimental Results Cont.

Viewing directions intersect			Initial Feasible Pose					Final Feasible Pose				
k =10 baseline =50			Harris Corner			Hough		Harris Corner			Hough	
Nodes	Edges	Actual(mm)	Total MSE	Measured	Error	Measured	Error	Total MSE	Measured	Error	Measured	Error
N0	E0	60		56.733591	5.44	68.663167	14.4		61.9789	3.3	63.3932	5.66
	E2	60	7.569004	64.348269	7.25	71.388108	19	0.602017	62.3801	3.97	63.7242	6.21
N1	E0	60		45.06763	24.9	49.828119	17		59.7165	0.47	59.554	0.74
	E2	60	3.956937	52.635919	12.3	44.351583	26.1	0.498929	59.6624	0.56	59.6755	0.54
N2	E0	60	7.808485	56.266426	6.22	62.178742	3.63	0.601031	59.7634	0.39	59.8792	0.2
N3	E2	60	6.014124	49.175597	18	40.819151	32	0.600634	59.5527	0.75	59.5912	0.68

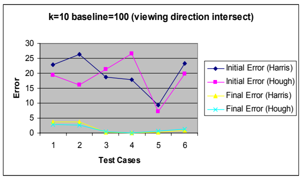
Viewing directions parallel			Initial Feasible Pose					Final Feasible Pose				
k =10 baseline =50			Harris Corner			Hough		Harris Corner			Hough	
Nodes	Edges	Actual(mm)	Total MSE	Measured	Error	Measured	Error	Total MSE	Measured	Error	Measured	Error
N0	E0	60		75.778099	26.3	57.190589	4.68		59.1682	1.39	58.7669	2.06
	E2	60	7.620014	45.318417	24.5	65.389527	8.98	0.801554	58.9877	1.69	58.7261	2.12
N1	E0	60		51.495489	14.2	54.762619	8.73		59.4453	0.92	61.2719	2.12
	E2	60	4.004354	56.054963	6.58	52.648415	12.3	1.306685	59.7688	0.39	59.9086	0.15
N2	E0	60	7.846992	55.316855	7.81	49.638213	17.3	0.853558	59.7631	0.39	60.1782	0.3
N3	E2	60	6.087683	63.79214	6.32	60.873159	1.46	0.853558	59.9355	0.11	60.1561	0.26

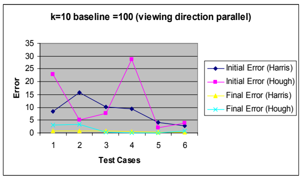
Experimental Results Cont.

Viewing directions parallel			Initial Feasible Pose					Final Feasible Pose				
k =10 baseline =100			Harris Corner			Hough		Harris Corner			Hough	
Nodes	Edges	Actual(mm)	Total MSE	Measured	Error	Measured	Error	Total MSE	Measured	Error	Measured	Error
N0	E0	60		64.953937	8.26	46.313933	22.8		59.5485	0.75	58.1275	3.12
	E2	60	7.152084	50.603244	15.7	63.087112	5.15	1.333199	60.4972	0.83	58.0204	3.3
N1	E0	60		53.882935	10.2	55.411536	7.65		60.4152	0.69	59.9221	0.13
	E2	60	3.809072	54.445302	9.26	77.223471	28.7	1.512935	59.7191	0.47	60.0602	0.1
N2	E0	60	7.312015	57.637242	3.94	61.206348	2.01	1.498624	59.7976	0.34	59.9936	0.01
N3	E2	60	5.799374	61.678789	2.8	57.653691	3.91	1.498622	59.8612	0.23	59.5088	0.82

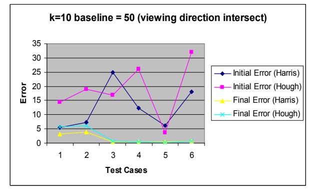
Viewing directions intersect			Initial Feasible Pose					Final Feasible Pose				
k =50 baseline =100			Harris Corner			Hough		Harris Corner			Hough	
Nodes	Edges	Actual(mm)	Total MSE	Measured	Error	Measured	Error	Total MSE	Measured	Error	Measured	Error
N0	E0	60		73.726807	22.9	51.559672	14.1		62.2296	3.72	61.72	2.87
	E2	60	9.479904	78.225027	30.4	74.446108	24.1	0.927276	61.9392	3.23	61.6125	2.69
N1	E0	60		53.246219	11.3	46.832184	21.9		59.8747	0.21	59.5302	0.78
	E2	60	4.943489	49.287989	17.9	44.117943	26.5	0.901055	59.6148	0.64	59.061	1.57
N2	E0	60	9.763616	54.360113	9.4	55.897924	6.84	1.190823	59.8524	0.25	59.6303	0.62
N3	E2	60	7.526288	46.004767	23.3	47.775266	20.4	1.189845	59.3892	1.02	59.4101	0.98

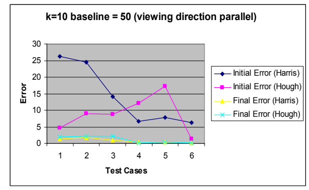
Experimental Results Cont.





Experimental Results Cont.

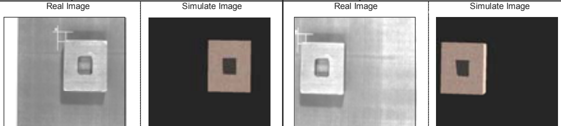




Experimental Results Cont.

Camera poses 1(3p)

Left camera pose: 31.901 44.077 612.709 1.584 -0.020 -3.086 | Right camera pose: 24.078 112.091 598.932 1.595 -0.026 -3.078

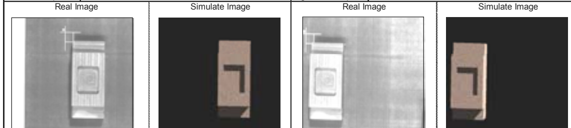


		Real Image			Simulate Image		
Edge:	Total RMSE:	Real Length	Measuerd Length	Error	Real Length	Measured Length	Error
E6	5.7075	50	50.1211	0.2422	50	49.815	0.37
E13	2.6897	50	50.4067	0.8134	50	50.0792	0.1584
E14	4.793	50	49.7604	0.4792	50	49.8823	0.2354
E2	3.9116	50	49.5708	0.8584	50	49.7427	0.5146
E39	13.609	16	16.4594	2.87125	16	16.112	0.7
E40	8.4911	16	16.0484	0.3025	16	15.8812	0.7425

Experimental Results Cont.

Camera poses 1(p2s)

Left camera pose: 31.901 44.077 612.709 1.584 -0.020 -3.086 | Right camera pose: 24.078 112.091 598.932 1.595 -0.026 -3.078

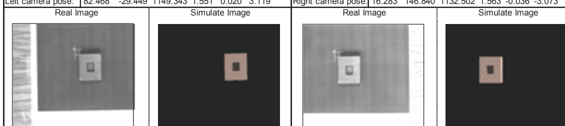


		Real Image			Simulate Image		
Edge:	Total RMSE:	Real Length	Measuerd Length	Error	Real Length	Measured Length	Error
E9	7.7583	38	37.9299	0.184473684	38	37.9269	0.1923684
E29	5.5643	38	38.7745	2.038157895	38	37.8946	0.2773684
E15	3.9007	52	52.2378	0.457307692	52	51.6928	0.5907692
E23	6.203	38	38.0313	0.082368421	38	38.2469	0.6497368
E18	6.0012	38	38.286	0.752631579	38	37.9324	0.1778947
E4	4.9396	52	53.038	1.996153846	52	51.9725	0.0528846
E26	5.5544	27	26.7687	0.856666667	27	27.0766	0.2837037
E25	8.555	25	24.664	1.344	25	24.8102	0.7592

Experimental Results Cont.

Camera poses 5(3p)

Left camera pose: 82.468 -29.449 1149.343 1.551 0.020 3.119 | Right camera pose: 16.283 146.840 1132.502 1.563 -0.036 -3.073

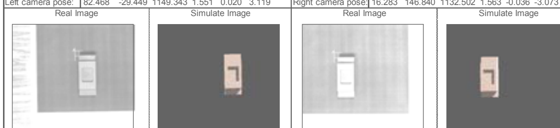


		Real Image			Simulate Image		
Edge:	Total RMSE:	Real Length	Measuerd Length	Error	Real Length	Measured Length	Error
E6	7.442	50	51.1405	2.281	50	51.9827	3.9654
E13	4.7417	50	50.5643	1.1286	50	50.8552	1.7104
E14	7.3688	50	51.1623	2.3246	50	49.5655	0.869
E2	4.9359	50	49.1213	1.7574	50	50.8061	1.6122
E39	39.8839	16	16.7707	4.816875	16	16.4107	2.566875
E40	26.2925	16	17.1159	6.974375	16	15.8119	1.175625

Experimental Results Cont.

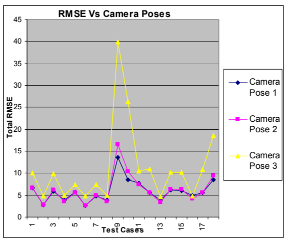
Camera poses 5(p2s)

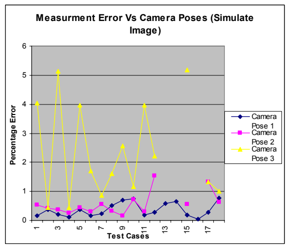
Left camera pose: 82.468 -29.449 1149.343 1.551 0.020 3.119 | Right camera pose: 16.283 146.840 1132.502 1.563 -0.036 -3.073



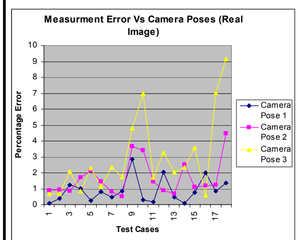
		Real Image			Simulate Image		
Edge:	Total RMSE:	Real Length	Measuerd Length	Error	Real Length	Measured Length	Error
E9	10.4795	38	38.6504	1.711578947	38	39.5098	3.9731579
E29	10.9313	38	39.2431	3.271315789	38	38.8457	2.2255263
E15	4.6844	52	53.0716	2.060769231	52		
E23	10.1762	38	38.8859	2.331315789	38		
E18	10.2699	38	39.3616	3.583157895	38	39.9721	5.1897368
E4	4.8594	52	52.3182	0.611923077	52		
E26	10.8504	27	25.0996	7.038518519	27	27.3609	1.3366667
E25	18.6123	25	27.2828	9.1312	25	25.2505	1.002

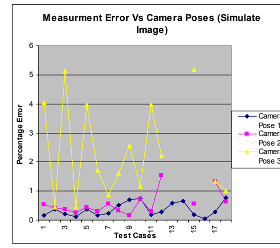
Experimental Results Cont.





Experimental Results Cont.





Conclusion

- Relative mean square error due to displacement and quantization is formulated
  - Stereo Error Model
- Nonlinear program incorporated with visual constraints, stereo system, and RMSE
  - Initial feasible solution
- Determine Optimal Camera pose
- Convert AG to EAG
  - Introduce the merging of viewing regions
- Automatic Visual Inspection Planning System