Flexible Automated Visual Inspection Planning Framework using Stereo Sensor with Error Reduction --Proposal

By Alexis H Rivera

Outline

- **Introduction and Motivation**
- Existing Related Works
- Problem Formulation and Statement
- Solution Approach
- Details
- Comparison with Related Work
- Plan for Work

Automated Inspection

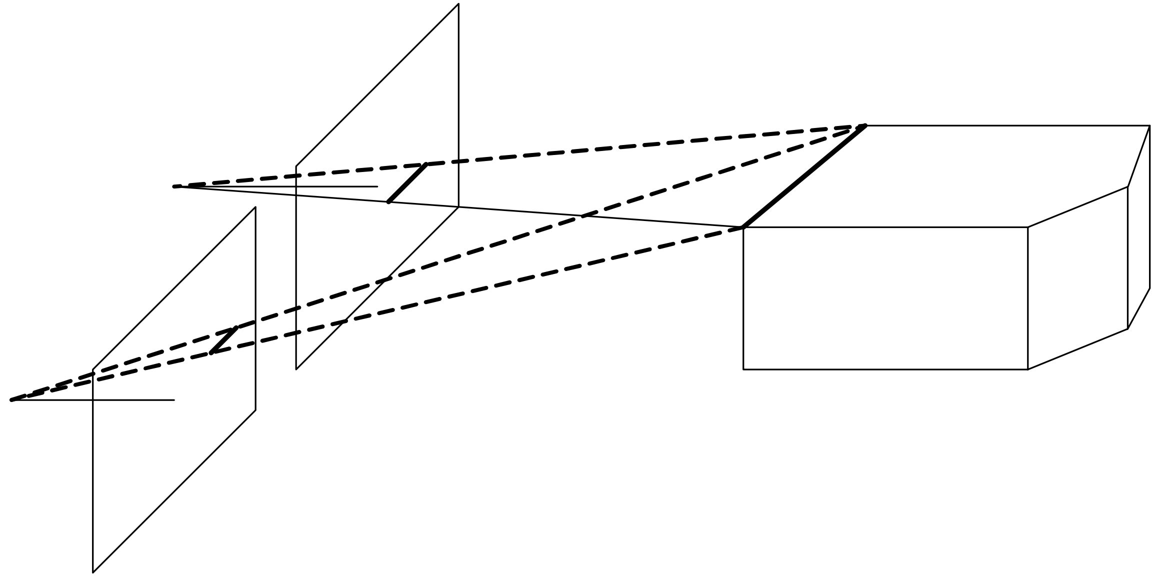
- used in manufacturing industry for quality control tasks
- tireless; relatively error free, low operating costs compared to human
- use of non contact sensors (cameras) reduces risk of damage during inspection
- use of cameras as sensors => automated visual inspection

Types of visual inspection systems

- Four types (Malamas, E. N., et al. (2003). "A Survey on Industrial Vision Systems, Applications, and Tools."):
  - dimensional quality
    - Are the dimensions of the object within the specified tolerances?
  - surface quality
    - Are there cracks in the object's surface?
  - correct assembling (structural quality)
    - Are there missing components on the PCB board?
  - correct operation (operational quality)
    - Is the conveyor belt moving as specified?

Introduction

- This research deals with the dimensional inspection of the edges of polyhedral objects using a stereo camera as the sensor



Automated Visual Inspection Planning

- Steps in automated visual inspection system design process (Mason, S. O. and A. Grun (1995). "Automatic Sensor Placement for Accurate Dimensional Inspection.")
  - Zero order design: specifying what to measure
  - First order design: selecting optimal network of cameras
  - Second order design: selecting the measurement precision
  - Third order: network densification

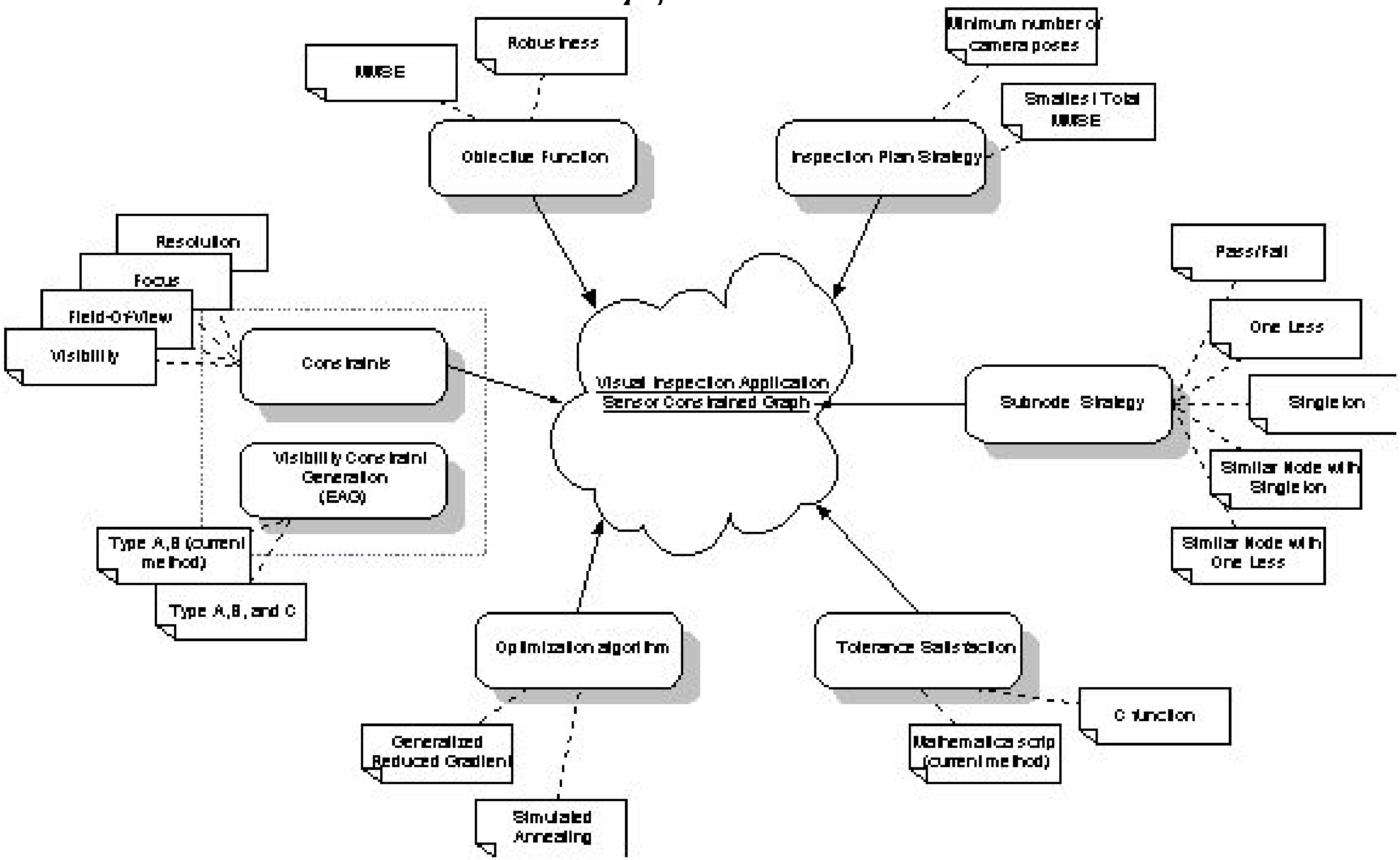
Automated Visual Inspection Planning

- Drawbacks:
  - Network design can be tedious and is usually accomplished by domain experts
- Automation can result in greater flexibility and lower costs

Automated Visual Inspection Planning

- This research addresses the ZOD, FOD, and SOD
- ZOD: Measuring object edges
- FOD: Finding optimal camera positions using nonlinear and integer optimization programs
- SOD: Minimizing the MSE of the line length

Flexible Automated Visual Inspection Planning Framework



Outline

- Introduction and Motivation
- **Existing Related Works**
- Problem Formulation and Statement
- Solution Approach
- Details
- Comparison with Related Work
- Plan for Work

Existing work

- Previous Automated Visual Inspection Planning
- Error Models and Sensor Constraints
- Visibility Representations:
  - Aspect Graphs, Entity-Based Aspect Graphs, Sensor Constraint Graphs
- Photogrammetry and Multiple View Geometry

Existing work

- Previous Automated Visual Inspection Planning
  - Crosby, K. (1997). Visual Inspection Planning with Error Reduction (VIPER).
  - Tarabanis, K. A., R. Y. Tsai, et al. (1995). "The MVP sensor planning system for robotic vision tasks."
  - Abrams, S., P. Allen, et al. (1999). "Computing Camera Viewpoints in an Active Robot Workcell."
  - Cowan, C. K. and P. D. Kovesi (1988). "Automatic Sensor Placement from Vision Task Requirements."
  - Mason, S. O. and A. Grun (1995). "Automatic Sensor Placement for Accurate Dimensional Inspection."
  - Olague, G. and R. Mohr (2002). "Optimal camera placement for accurate reconstruction."
  - Sakane, S., M. Sato, et al. (1990). "Automatic Planning of Light Source Placement for an Active Photometric Stereo System."
  - Solomon, F. and K. Ikeuchi (1995). "An Illumination Planner for Lambertian Polyhedral Objects."
  - Yi, S., R. M. Haralick, et al. (1990). "Automatic Sensor and Light Source Positioning for Machine Vision."

Existing work

- Error Models, Sensor Constraints
  - Yang, C. C., M. M. Marefat, et al. (1999). "Modeling Errors for Dimensional Inspection Using Active Vision."
  - Gu, X., M. Marefat, et al. (1999). "A Robust Approach for Sensor Placement in Automated Vision Dimensional Inspection."
  - Tarabanis, K., R. Y. Tsai, et al. (1994). "Analytical characterization of the feature detectability constraints of resolution, focus, and field- of-view for vision sensor planning."
  - Rodriguez, J. J. and J. K. Aggarwal (1990). "Stochastic Analysis of Stereo Quantization Error."
  - Zhao, W. and N. Nandhakumar (1996). "Effects of Camera Alignment Errors on Stereoscopic Depth Estimates."
  - Cooper, M. A. R. and P. A. Cross (1988). "Statistical concepts and their application in photogrammetry and surveying."
  - Hartley, R. and P. Sturm (1997). "Triangulation."

Existing work

- Aspect Graphs, EAG, SCG
  - Stewman, J. H. (1991). Viewer-centered representations for polyhedral objects: computing the exact perspective projection aspect graph of an object bounded by planar faces
  - Yang, C. C., M. M. Marefat, et al. (1998). "Entity-Based Aspect Graph: Making Viewer Centered Representations More Efficient."
  - Crosby, K. (1997). Visual Inspection Planning with Error Reduction (VIPER).
- Union Polyhedra, max volume ellipsoid
  - Bemporad, A., K. Fukuda, et al. (2000). Convexity Recognition of the Union of Polyhedra
  - Zhang, Y. (1998). An Interior-Point Algorithm for the Maximum-Volume Ellipsoid Problem

Existing work

- Photogrammetry and Multiple View Geometry
  - Hartley, R. and A. Zisserman (2000). Multiple view geometry in computer vision.
  - Faugeras, O. (1993). Three-dimensional computer vision : a geometric viewpoint.
  - Fraser, C. S. (1984). "Network Design Considerations for Non-Topographic Photogrammetry."
  - Triggs, B., P. F. McLauchlan, et al. (2000). Bundle Adjustment - A Modern Synthesis.

Outline

- Introduction and Motivation
- Existing Related Works
- **Problem Formulation and Statement**
- Solution Approach
- Details
- Comparison with Related Work
- Plan for Work

Problem Formulation (Part 1)

Let

E be the set of entities of interest $(e_1,e_2,\ldots)$

P be the set of optimal camera poses $(p_1,p_2,\ldots)$

S_i be a tuple that associates a camera pose p_i with a subset E_i of entities of interest

F be an objective function to minimize

T_i be the set of specified tolerances of the entities e_i

Th_i be a threshold for the acceptability of the measurement for the entities e_i

We want to find a set S such that:

$S = \cup S_i \; \forall i$ such that

$\cup \text{entities}(S) = E$

F(S) is minimized

$\text{Prob}(\text{dimensional\_error}(e_j,p_j) < T_j) >= Th_j \; \forall e_j \in \text{entities}(S_i), \forall p_j \in \text{pose}(S_i), \forall S_i \in S$

Problem Formulation (Part 2)

- Optimal camera pose as nonlinear program

<u>Input</u>
Set of entities of interest E
F is a function that defines the optimality criterion.
This function takes as a parameter the entities of interest S and the camera pose (t_x, t_y, t_z, Φ, θ, Ψ).

<u>Output</u>
Optimal camera pose (t_x, t_y, t_z, Φ, θ, Ψ) and optimal value

<u>Algorithm</u>
Minimize F (t_x, t_y, t_z, Φ, θ, Ψ, E)
Subject to:
g1j <= 0 (resolution), for j=1 to k, where k is the number of entities in E
g2a <= 0 (focus)
g2b <= 0 (focus)
g3 <= 0(field of view)
g4 <= 0 (incidence angle)
g5 <= 0 (room size)
g6i <= 0 (visibility) for i=1 to m, where m is the number of hyperplanes that define the visibility boundary

Problem Formulation (Part 3)

- F() is a metric that relates the camera pose to the expected accuracy of the measurement

Outline

- Introduction and Motivation
- Existing Related Works
- Problem Formulation and Statement
- **Solution Approach**
- Details
- Comparison with Related Work
- Plan for Work

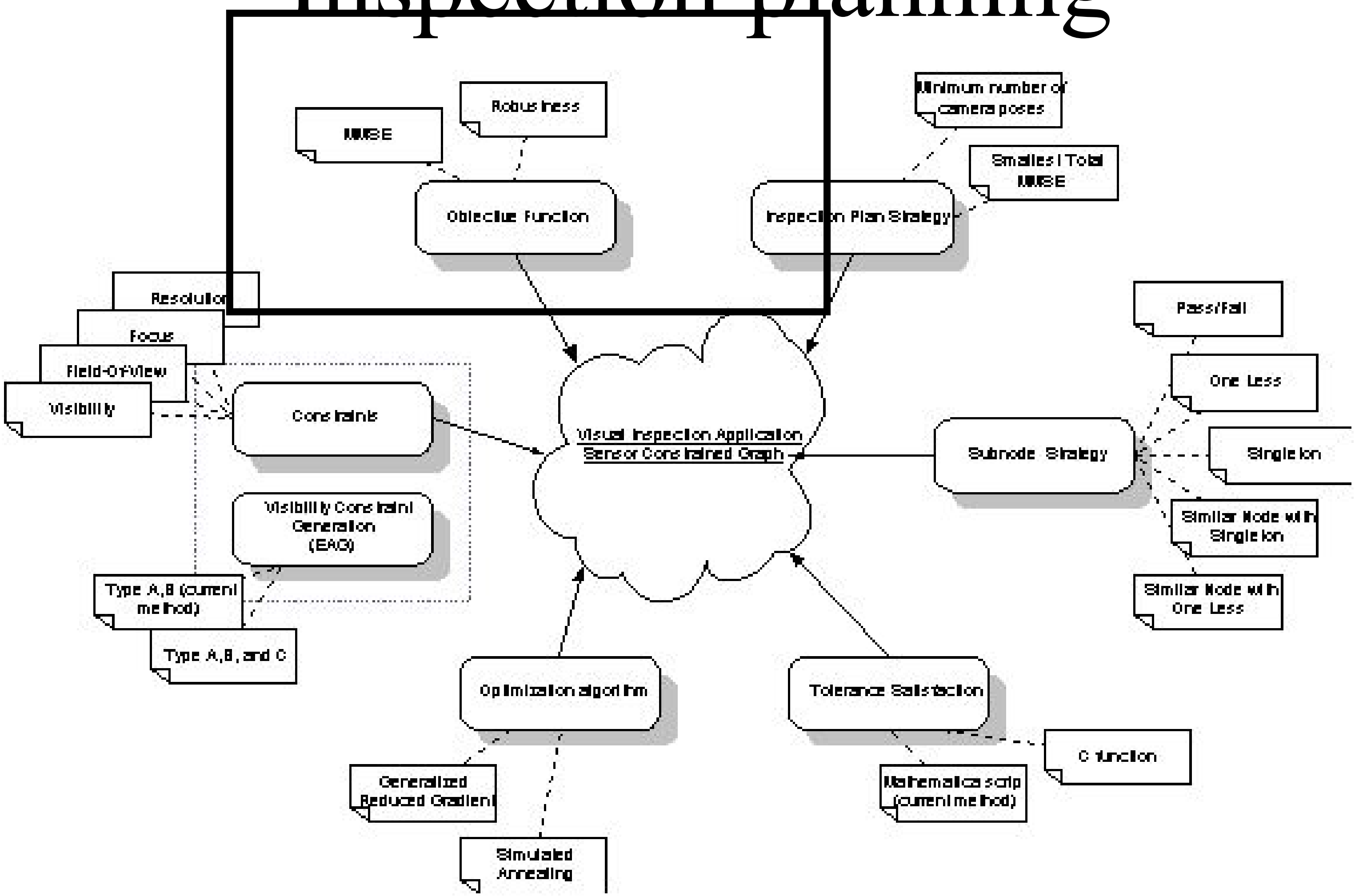
Solution approach

- Define stereo error model and objective function
- Summary of sensor constraints
- Summary nonlinear program
- Summary inspection planning algorithm
  - Sensor Constraint Graph
  - Integer Program
  - Tolerance evaluation

Outline

- Introduction and Motivation
- Existing Related Works
- Problem Formulation and Statement
- Solution Approach
- **Details**
- Comparison with Related Work
- Plan for Work

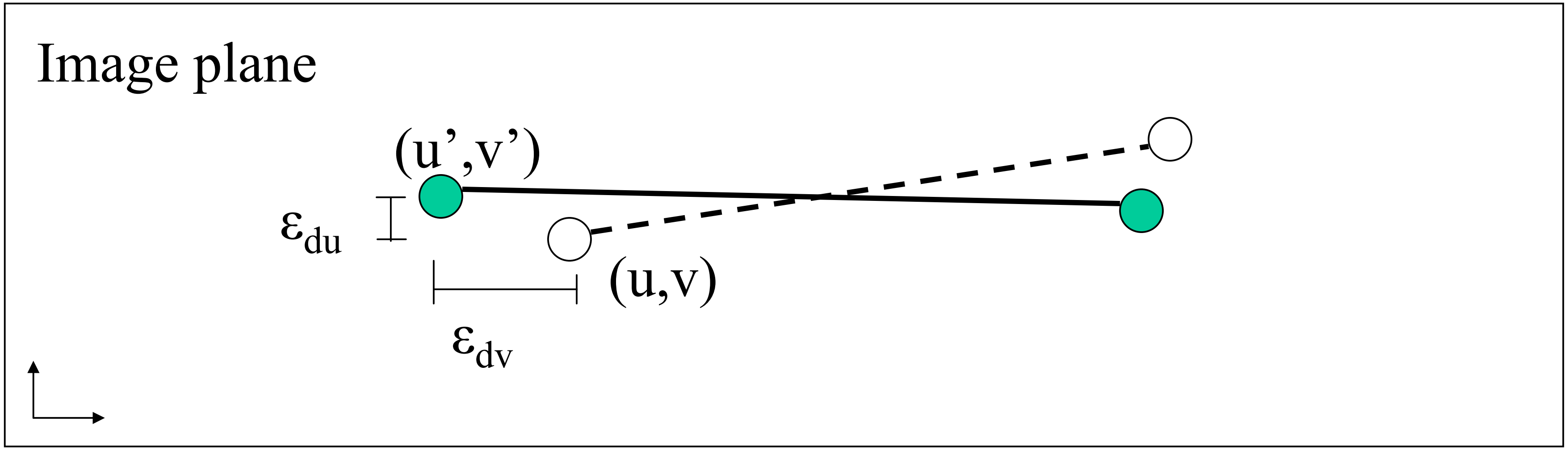
Inspection planning



Error models

- Previous work
  - Sources of error in camera sensor
  - Crosby's Mean Square Error of Displacement and Quantization Error
  - Crosby's Probability that Error is Within Specified Tolerances

Displacement error of single point

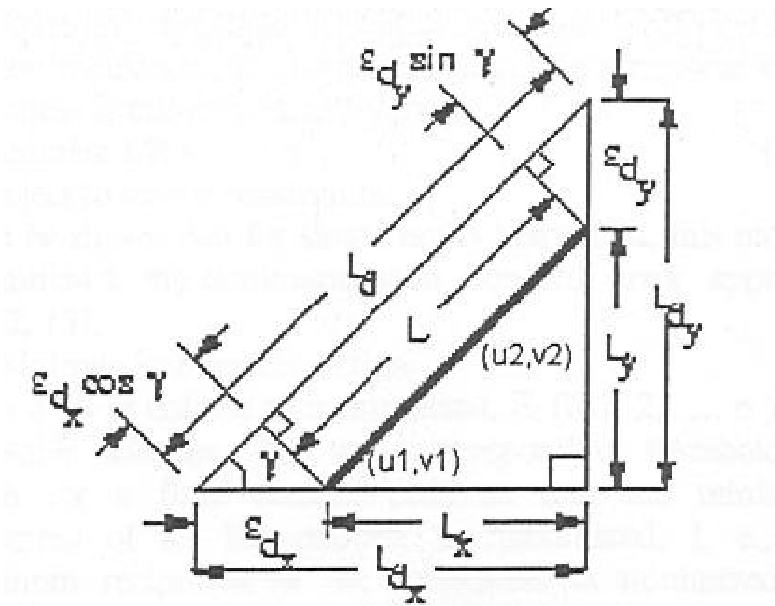


Displacement error for a each end point are new Gaussian RV

$\varepsilon_{du} = u' - u$

$\varepsilon_{dv} = v' - v$

Displacement error of line



- Displacement error is geometrically approximated:

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

  γ = angle between line

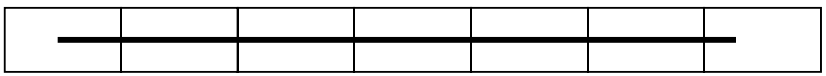
Displacement error of k lines

- Total dimensional error due to displacement for k lines is:

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

$$E[\varepsilon_d^2] = \sigma_{\varepsilon_d}^2 + \eta_{\varepsilon_d}^2$$

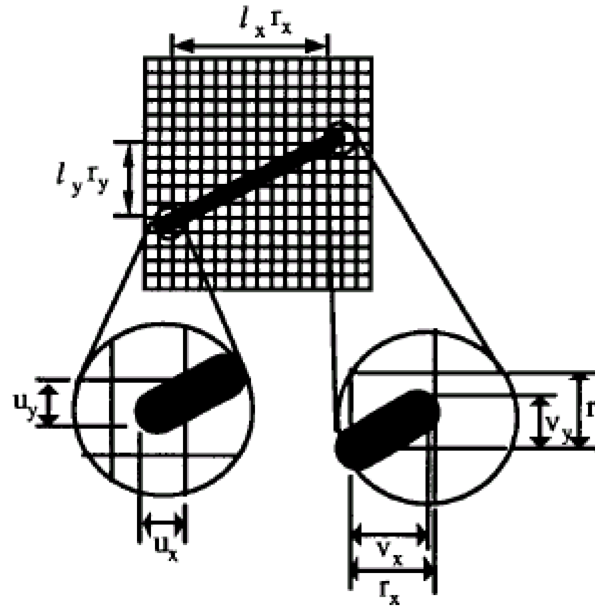
Quantization Error 1D



- Actual Length:

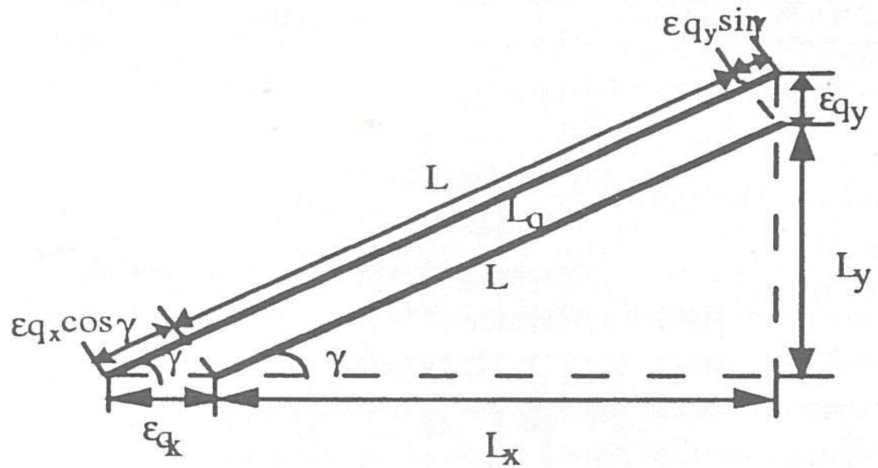
  L = $lr_x + u + v$, where u,v uniform random variables

- Quantized Length



$$Lq = \begin{cases} lr_x & u \le .5 \cap v \le .5 \\ (l+2)r_x & u > .5 \cap v > .5 \\ (l+1)r_x & (u \le .5 \cap v > .5) \cup (u > .5 \cap v \le .5) \end{cases}$$

Quantization Error for a line



- Total quantization determined by geometric approximation,

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

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

Total quantization error for k lines

Total dimensional error due to quantization in all lines:

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

$$E[\varepsilon_q^2] = \sigma_{\varepsilon_q}^2 = \sum_{j=1}^{k} \sigma_{\varepsilon_q j}^2$$

Dimensional Tolerances

- Dimensional Tolerance is satisfied if

$$\int_{-\Delta L}^{\Delta L} f_{\varepsilon}(\varepsilon)\,d\varepsilon \geq Threshold$$

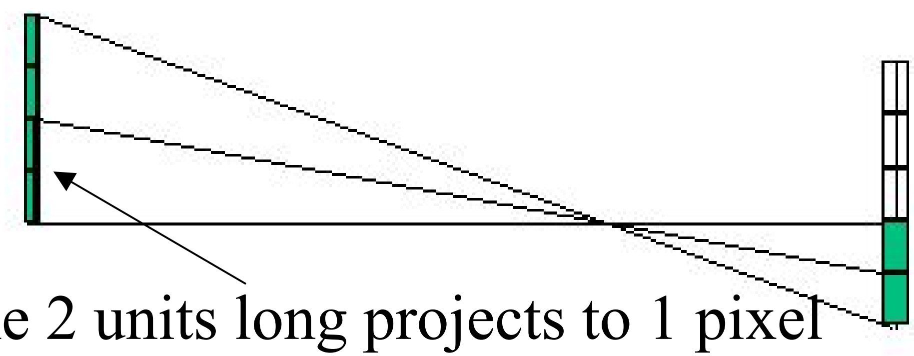
- $f_{\varepsilon}(\varepsilon)$ is the probability density function of dimensional inspection error

Error models

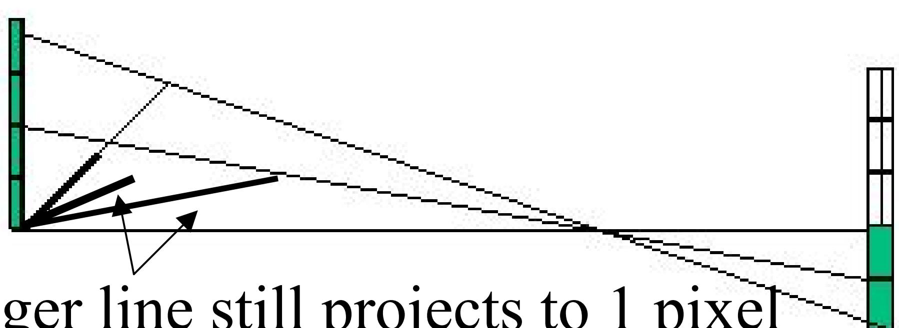
- What's wrong with Crosby's models?
- Why stereo?
- Sources of errors in stereo
- Stereo error approximation
  - least squares adjustment
  - Why least squares?

What’s wrong with Crosby’s models?

- Error in projected length of a single image does not relate to the 3D error



Line perpendicular to viewing direction, 3D length is scaling of projection



Lines at an angle can still project to the same length

Why stereo?

- At least 2 cameras needed to do 3D reconstruction
- It is easy to build a stereo system

Sources of errors in stereo

- Error sources include:
  - Quantization Error
    - due to spatial quantization in the CCD array
  - Localization Error
    - inaccuracies in the endpoint detection algorithms
  - Calibration Errors
  - Misalignment of cameras
- Introduce an error in the pixel location of the projected line endpoints

Least Square Error Estimation

- Functional model (for a single camera):
  - Let a_{ij} be rotation matrix coefficients from roll-pitch-yaw camera's rotation angles
  - Let X_o, Y_o, Z_o be the coordinates of the camera's perspective center
  - Let u,v be the measured image coordinate of a point X_1,Y_1,Z_1
  - Let c be the focal length
- $c[a_{11}(X_1-X_0)+a_{12}(Y_1-Y_0)+a_{13}(Z_1-Z_0)] - u[a_{31}(X_1-X_0)+a_{32}(Y_1-Y_0)+a_{33}(Z_1-Z_0)] = 0$
- $c[a_{21}(X_1-X_0)+a_{22}(Y_1-Y_0)+a_{23}(Z_1-Z_0)] - v[a_{31}(X_1-X_0)+a_{32}(Y_1-Y_0)+a_{33}(Z_1-Z_0)] = 0$

Least squares estimates

- Let **x** be the vector of elements whose values are to be found
  - $\mathbf{x} = [X_l, Y_l, Z_l, X_r, Y_r, Z_r]$
- Let **l** be the vector of elements which have been measured
  - $\mathbf{l} = [u_l, v_l, u_r, v_r]$
- Let **c** be the vector of elements whose values are known and regarded constant.
  - **c** = [rotation parameters, focal length, camera centers of both cameras]
- Let **f** be the vector of functional models
- Functional model can be summarized as
- $\mathbf{f}(\mathbf{x},\mathbf{l},\mathbf{c})=0$

Least squares estimate

Using subscripts l and r to denote left and right camera

$c[a_{11r}(X_1-X_{0r})+a_{12r}(Y_1-Y_{0r})+a_{13r}(Z_1-Z_{0r})] - u_r[a_{31r}(X_1-X_{0r})+a_{32r}(Y_1-Y_{0r})+a_{33r}(Z_1-Z_{0r})] = 0$

$c[a_{21r}(X_1-X_{0r})+a_{22r}(Y_1-Y_{0r})+a_{23r}(Z_1-Z_{0r})] - v_r[a_{31r}(X_1-X_{0r})+a_{32r}(Y_1-Y_{0r})+a_{33r}(Z_1-Z_{0r})] = 0$

$c[a_{11l}(X_1-X_{0l})+a_{12l}(Y_1-Y_{0l})+a_{13l}(Z_1-Z_{0l})] - u_l[a_{31l}(X_1-X_{0l})+a_{32l}(Y_1-Y_{0l})+a_{33l}(Z_1-Z_{0l})] = 0$

$c[a_{21ll}(X_1-X_{0l})+a_{22l}(Y_1-Y_{0l})+a_{23l}(Z_1-Z_{0l})] - v_l[a_{31l}(X_1-X_{0l})+a_{32l}(Y_1-Y_{0l})+a_{33l}(Z_1-Z_{0l})] = 0$

Least squares estimation

Let x,l be the true values of x,l

Let x_o, l_o be the first order approximation

Calculating first order approximation of f(x,l,c)

$$f(x_o,l_o,c) + df(\mathit{x}-x_o)/dx + df(\mathit{l}-l_o)/dl = 0$$

Let $b=-f(x_o,l_o,c)$

Let A = df/dx, B=df/dl

Let $x = \mathit{x}-x_o$, $v=\mathit{l}-l_o$

Solving Ax+Bv=b for x and v, give the corrections that will give the estimated vector *x* and *l*

Least square estimation

- Assume measurement vector l of m measurements has the following covariance matrix
- $C_l = diag[\sigma_1^2, \sigma_2^2, .. \sigma_m^2]$
- Define σ_o^2 as the reference variance
- Define the weight matrix $W = \sigma_o^2 inv(C_l)$
- Define the cofactor matrix $Q_l = inv(W)$

Least squares estimate

- Solution to Ax+Bv = b

$$\hat{x} = \left[A^T \left(B W^{-1} B^T \right)^{-1} A \right]^{-1} A^T \left(B W^{-1} B^T \right)^{-1} b$$

$$\hat{k} = \left(B W^{-1} B^T \right)^{-1} \left(A\hat{x} - b \right)$$

$$\hat{v} = -W^{-1} B^T \hat{k}$$

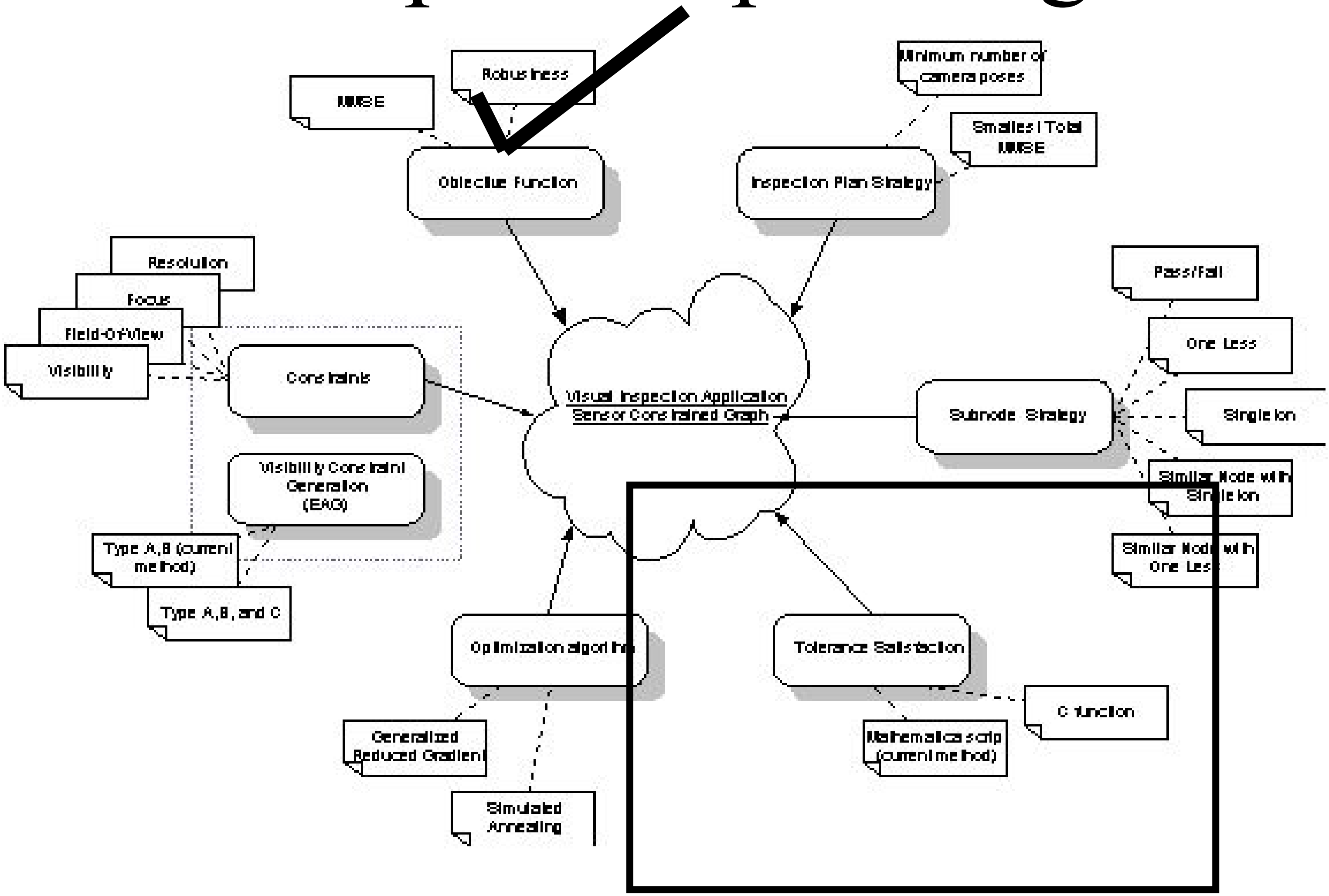
- Cofactor matrix of estimated positions x

$$Q_{\hat{x}} = \left[A^T \left(B Q_l B^T \right)^{-1} A \right]^{-1}$$

Why least squares?

- From M.A.R. Cooper, P.A. Cross, (1988), “Statistical Concepts and Their Application in Photogrammetry and Surveying”
  - Simple to estimate, linear estimate
  - unique
  - unobjectionable (it is not easy to find an argument against using it)
  - leads to a simple quantitative assessment of quality
  - least squares estimates are unbiased
  - describe a minimum variance estimate
  - best linear unbiased estimate independent of PDF of measurement errors
  - if PDF is normal ==> maximum likelihood estimate

Inspection planning



MSE of the 3D length of a line

Let $\varepsilon_{pi} = X_i - X_i'$ (random vector representing the error of estimated 3D point X_i')

Assume ε_{pi} is Normal with $E[\varepsilon_{pi}]=0$, $Cov(\varepsilon_{pi})=C_i$ (from the LSE algortithm)

Let ε_{p1}, ε_{p2} be the errors of points X_1 and X_2

Let $\varepsilon_d = \varepsilon_{p1} - \varepsilon_{p2}$ be the dimensional error vector of the line formed by X_1 and X_2

Let u be the unit vector vector representing the direction of the line

Then, the statistics of the dimensional error for a single line can be derived as follows:

$$E\left[\varepsilon_d\right]=0$$

$$J_{\varepsilon_d}=\begin{bmatrix}\frac{\partial\varepsilon_d}{d\varepsilon_{p1}} & \frac{\partial\varepsilon_d}{d\varepsilon_{p2}}\end{bmatrix}=\begin{bmatrix}1 & 0 & 0 & -1 & 0 & 0\\ 0 & 1 & 0 & 0 & -1 & 0\\ 0 & 0 & 1 & 0 & 0 & -1\end{bmatrix}$$

$$C_{\varepsilon_d}=J^T_{\varepsilon_d}\begin{bmatrix}C_1 & 0\\ 0 & C_2\end{bmatrix}J_{\varepsilon_d}$$

$$\left\|\varepsilon_d\right\| \approx \varepsilon_d \cdot u$$

$$E\left[\left\|\varepsilon_d\right\|\right] = 0$$

$$J_{\left\|\varepsilon_d\right\|} = \frac{\partial\left\|\varepsilon_d\right\|}{d\,\varepsilon_d} = \begin{bmatrix}u_x & u_y & u_z\end{bmatrix}$$

$$Var(\varepsilon_d) = J^T_{\left\|\varepsilon_d\right\|} C_{\varepsilon_d} J_{\left\|\varepsilon_d\right\|}$$

$$E[\varepsilon_d^2] = Var(\varepsilon_d)$$

Tolerance satisfaction

Recall ε_d is Gaussian with $E[\varepsilon_d]=0$ $Var[\varepsilon_d]=\sigma^2$

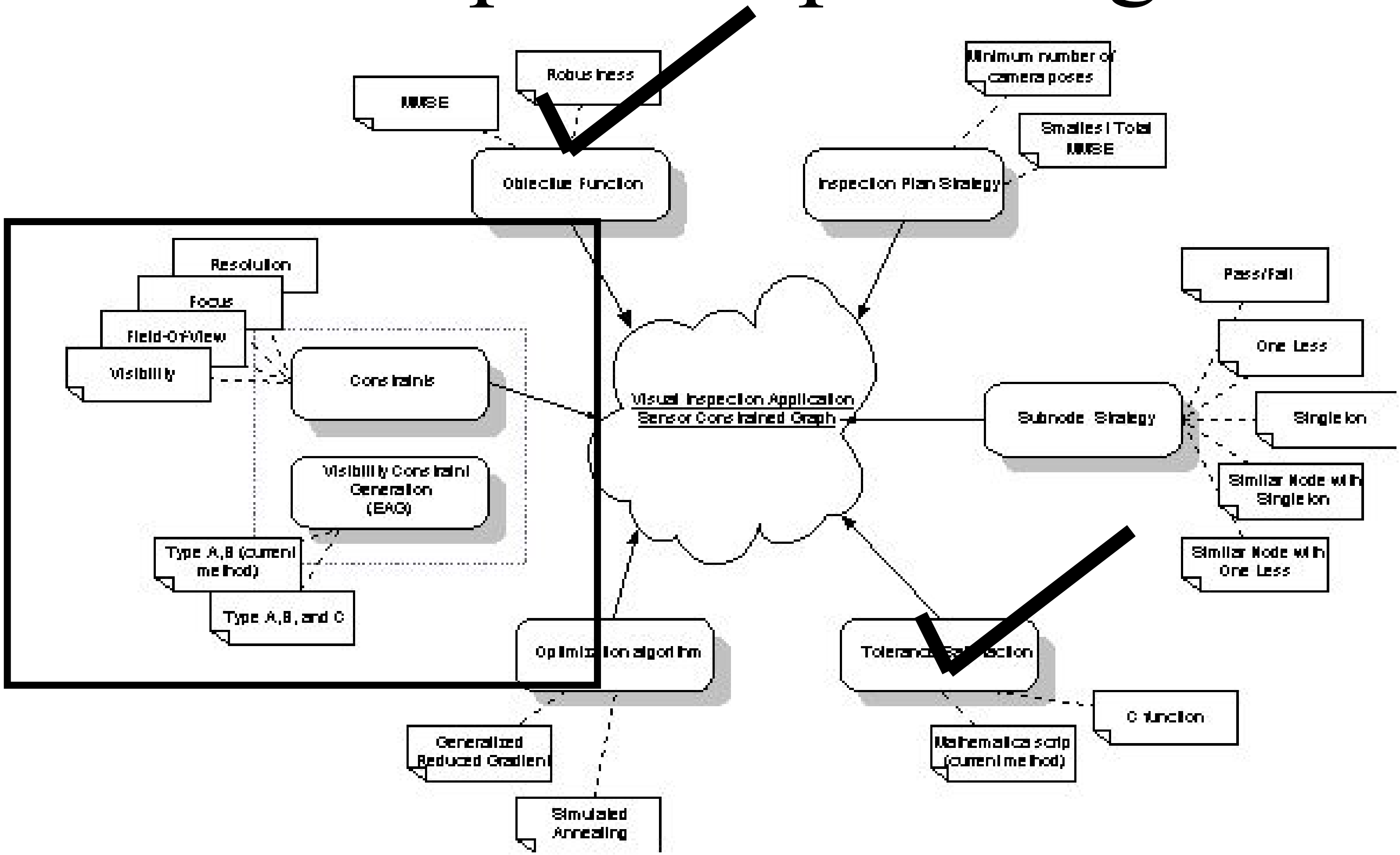
Let Δ_i be a tolerance specification for line i

For a 3D line of length L_i, the probability that the error is within the specified tolerance is:

- $Prob(-\Delta L <= \varepsilon_d <= \Delta L) = CDF(\Delta L, 0, \sigma)-CDF(-\Delta L,0, \sigma)$
- $CDF(x,\mu,\sigma)$ is the gaussian cumulative density function evaluated at x, with mean and std. dev. μ and σ respectively

- Accept the camera pose if
- $Prob(-\Delta L <= \varepsilon_d <= \Delta L) >=$ Threshold

Inspection planning



Nonlinear optimization program

Input
Set of entities of interest E
F is a function that defines the optimality criterion.
This function takes as a parameter the entities of interest S and the camera pose (t_x, t_y, t_z, Φ, θ, Ψ).

Output
Optimal camera pose (t_x, t_y, t_z, Φ, θ, Ψ) and optimal value

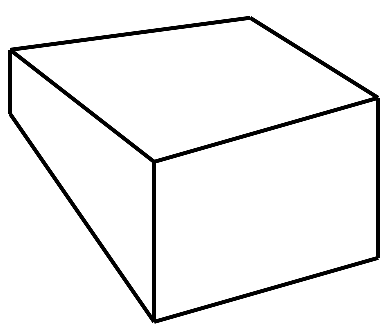
Algorithm
Minimize F (t_x, t_y, t_z, Φ, θ, Ψ, E)
Subject to:
g1j <= 0 (resolution), for j=1 to k, where k is the number of entities in E
g2a <= 0 (focus)
g2b <= 0 (focus)
g3 <= 0(field of view)
g4 <= 0 (incidence angle)
g5 <= 0 (room size)
g6i <= 0 (visibility) for i=1 to m, where m is the number of hyperplanes that define the visibility boundary

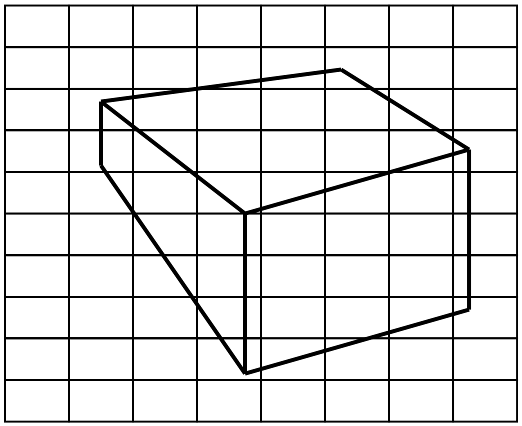
Sensor Constraints

- Resolution
- Focus
- Field of View
- Incidence Angle
- Room Size
- Visibility
  - Determining viewing volumes
  - Union of viewing volumes

Resolution

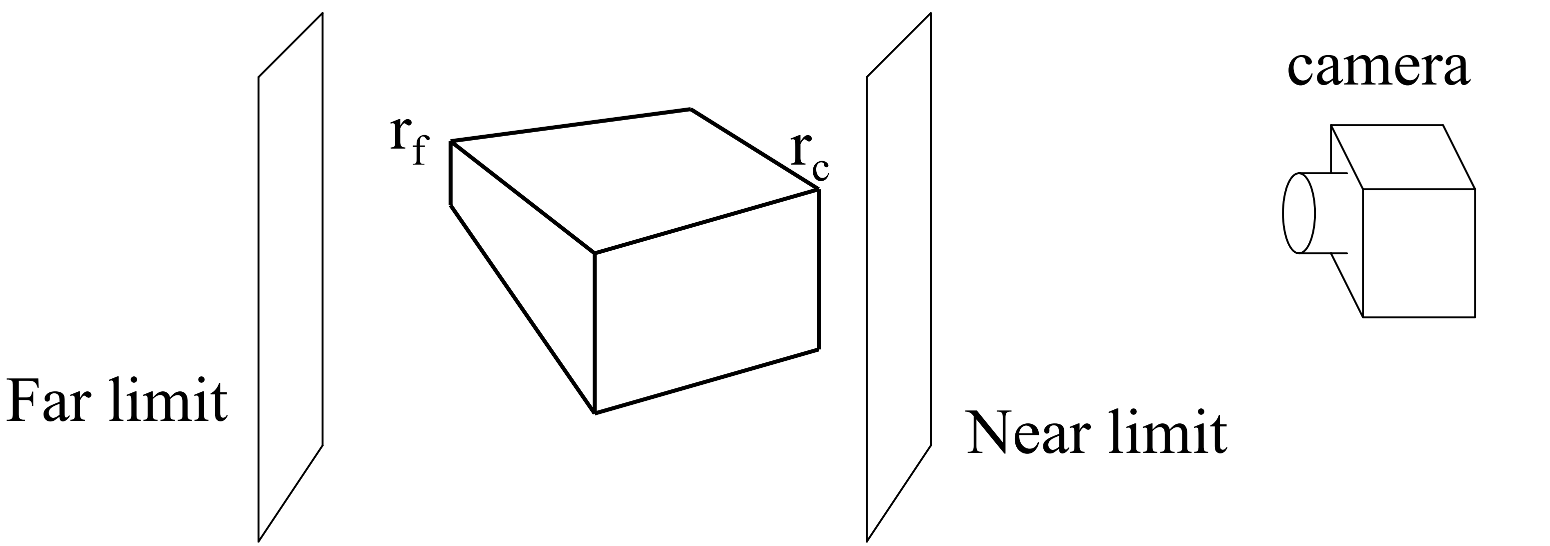
- For each entity j, there is a constraint g1j()
- Projects a line of *l* millimeters to a line of w millimeters





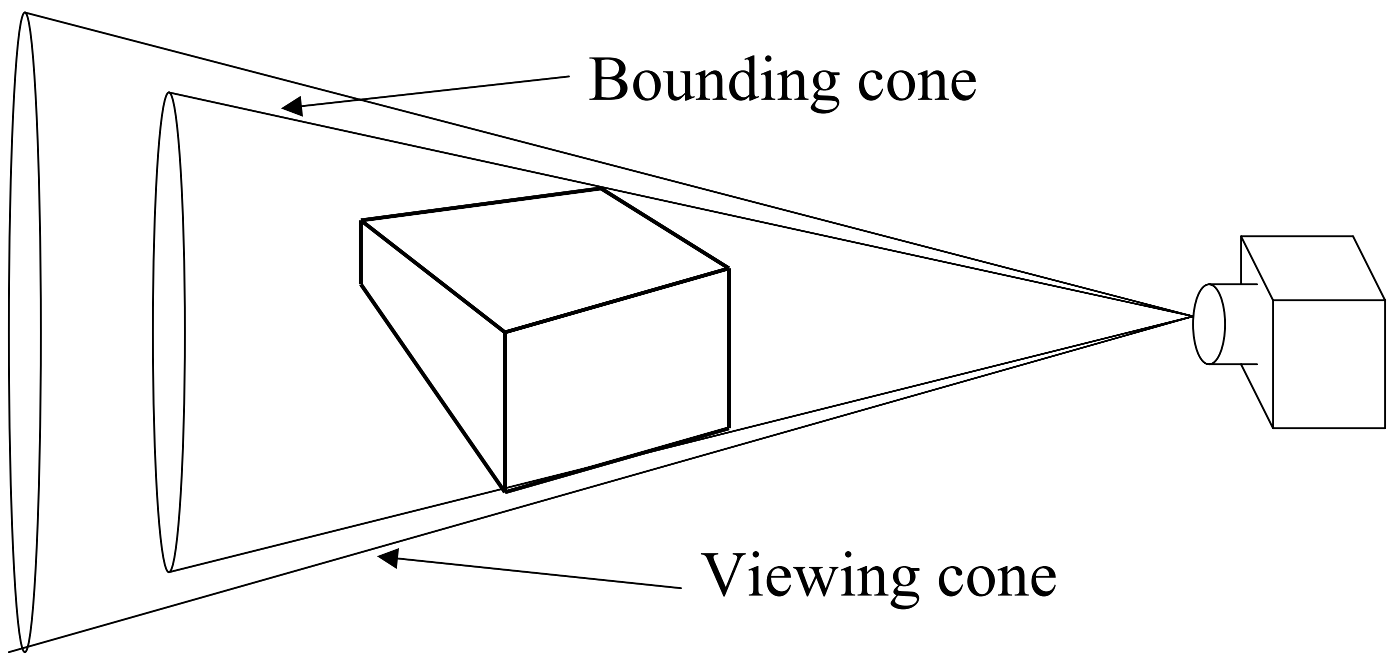
Focus

- Two constraints, g2a(), g2b()
- Require closest and furthest entity vertices from the camera position to be within the far and near limits of the depth of field



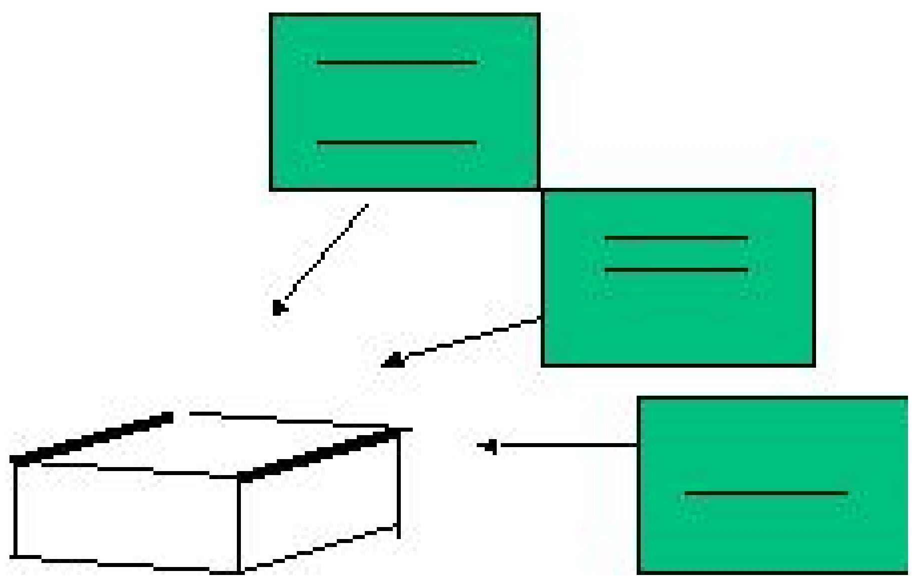
Field Of View

- One constraint: g3()
- Bounding cone must be contained within the viewing cone



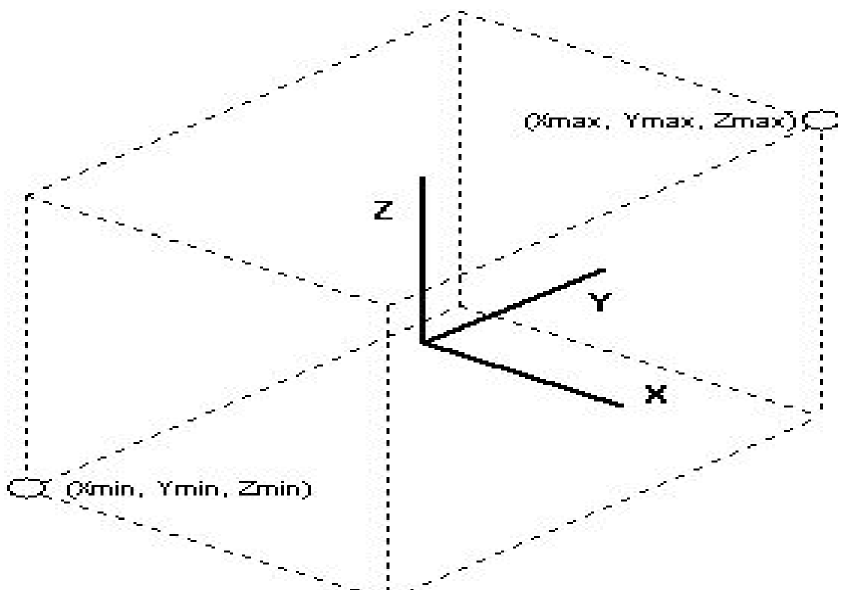
Incidence Angle

- Number of constraints depend on number of entities: g4()
- The incidence angle constraint prevents the camera position from being coplanar to the entities of interest



Room size

- Only one: g5()
- The room size constraint limits the possible range of camera positions.

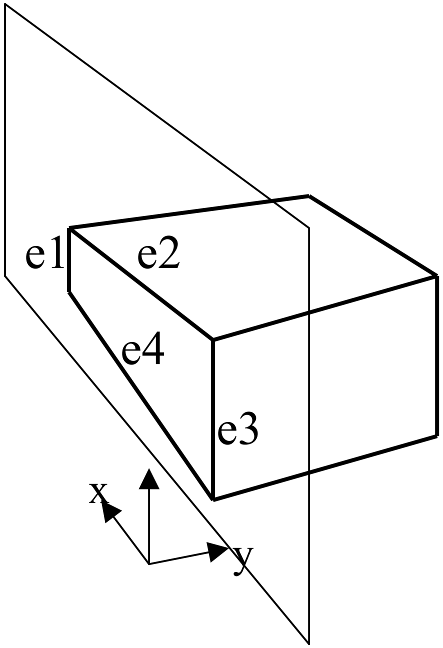


Visibility

- Many equations: g6i() for i=1 to m
- Plane equations that bound the visibility of the desired entities

Example:

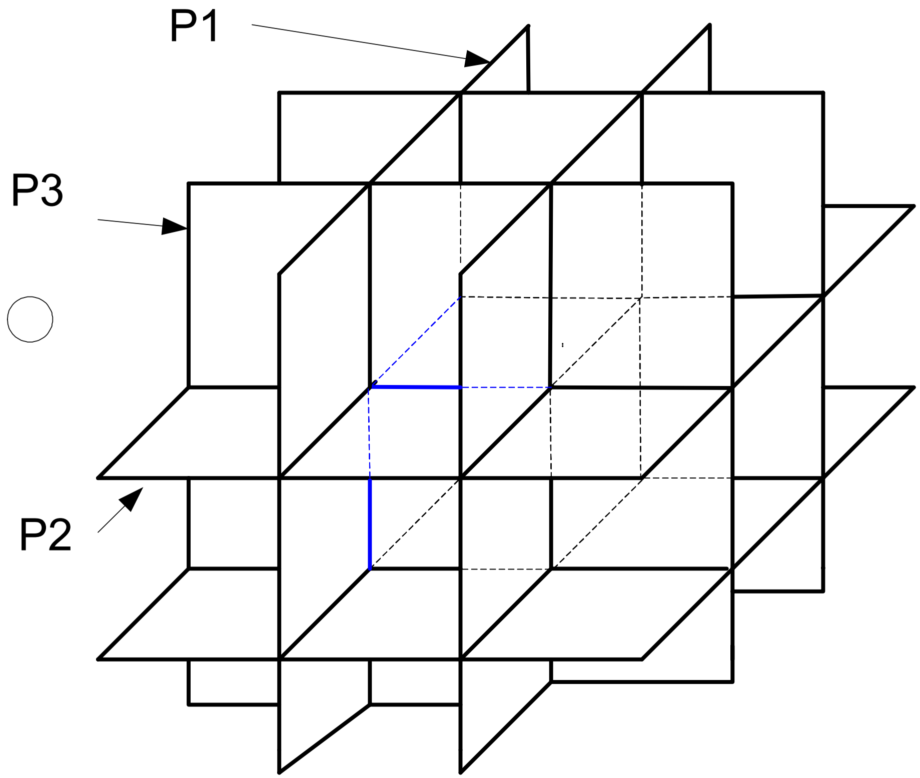
To see entities e1, e2, e3, e4, the camera must satisfy equation $y < 0$



Aspect Graph (Stewman)

- Viewing volumes as the intersection of hyperplanes
- Viewing volumes for all the entities in the object

Ex:



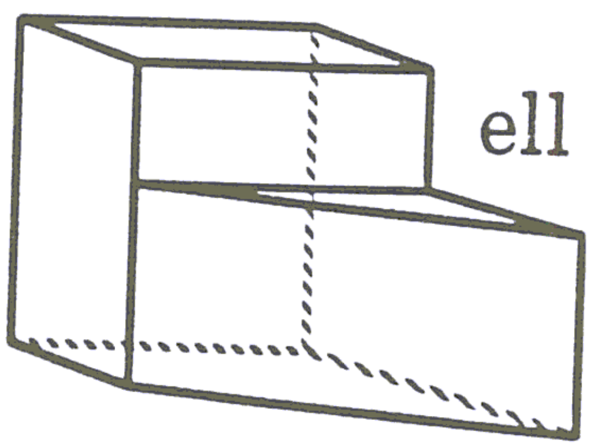
Viewing Volume:

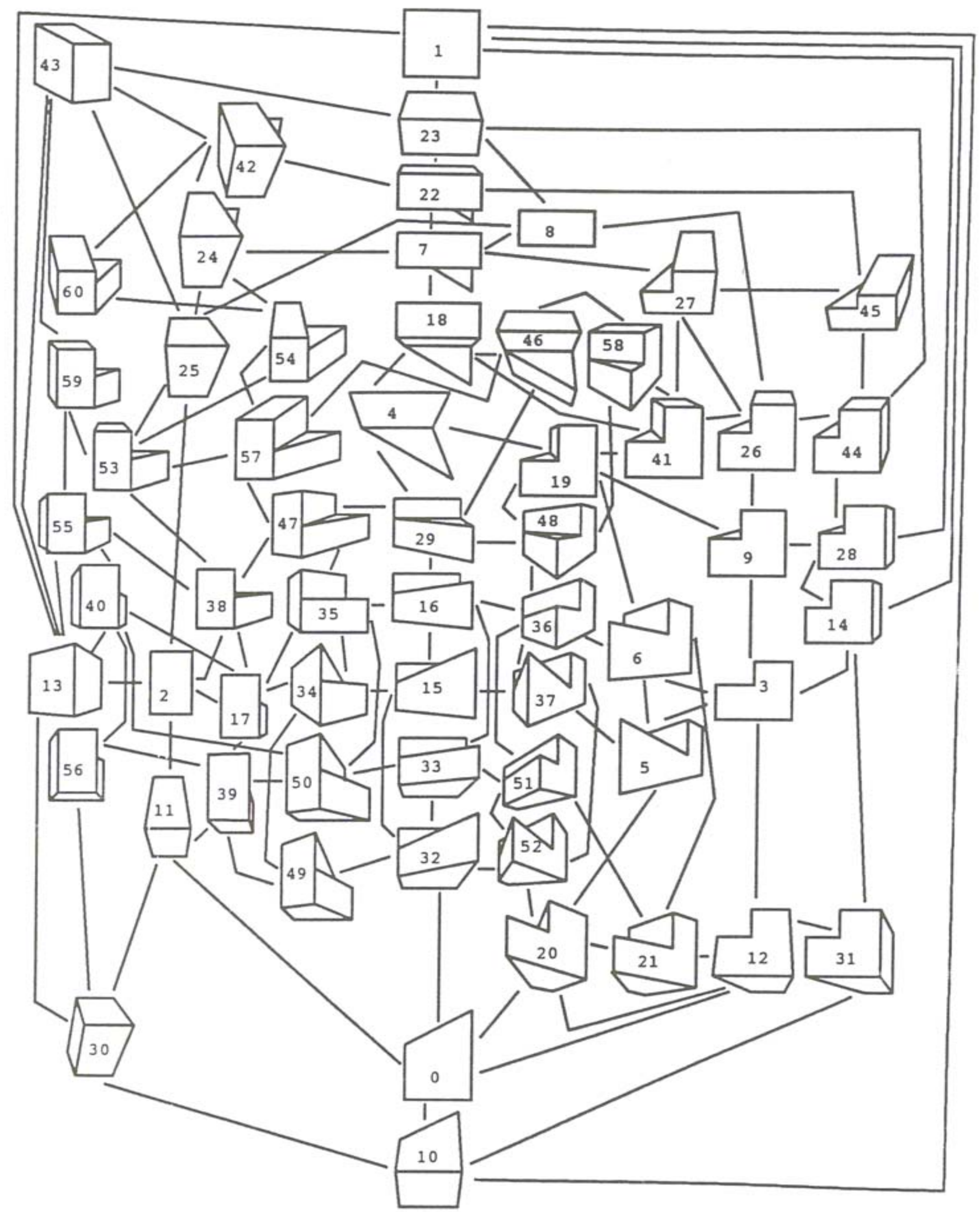
H1H2H3

•Drawbacks – may be too much information

Aspect Graph (Stewman) Cont.

- Aspect graph of ell





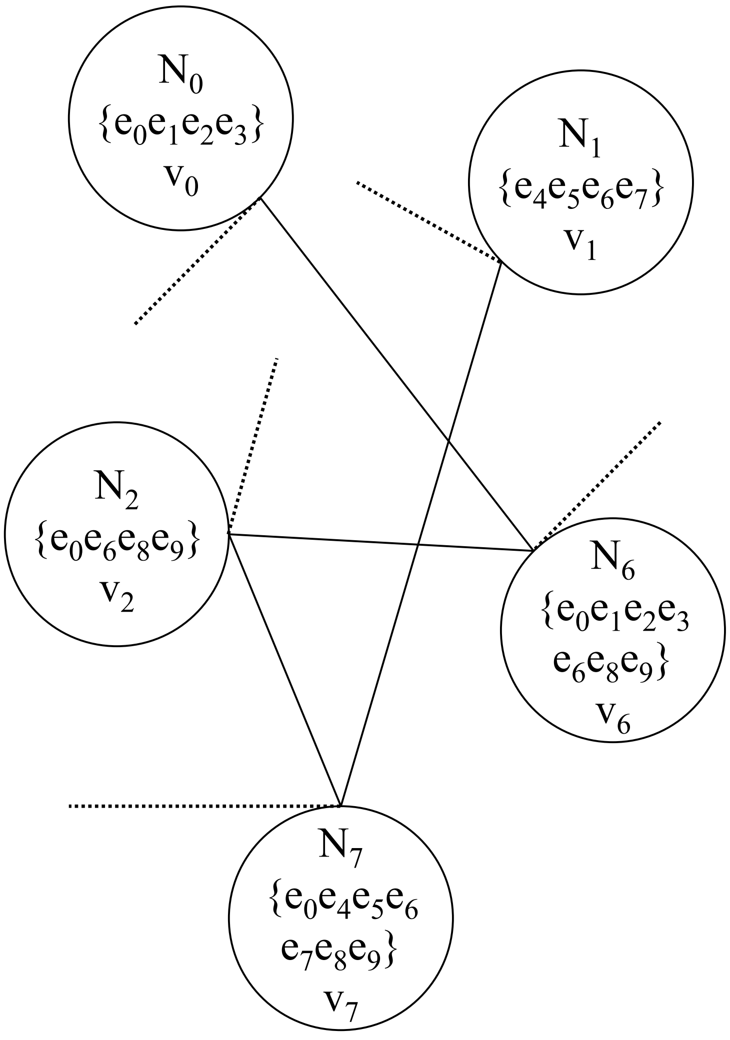
Entity Aspect Graph (Yang)**

- EAG simplifies the Aspect Graph, reducing the number of nodes
- Viewing regions for a subset of entities on the object
- 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

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 Example:

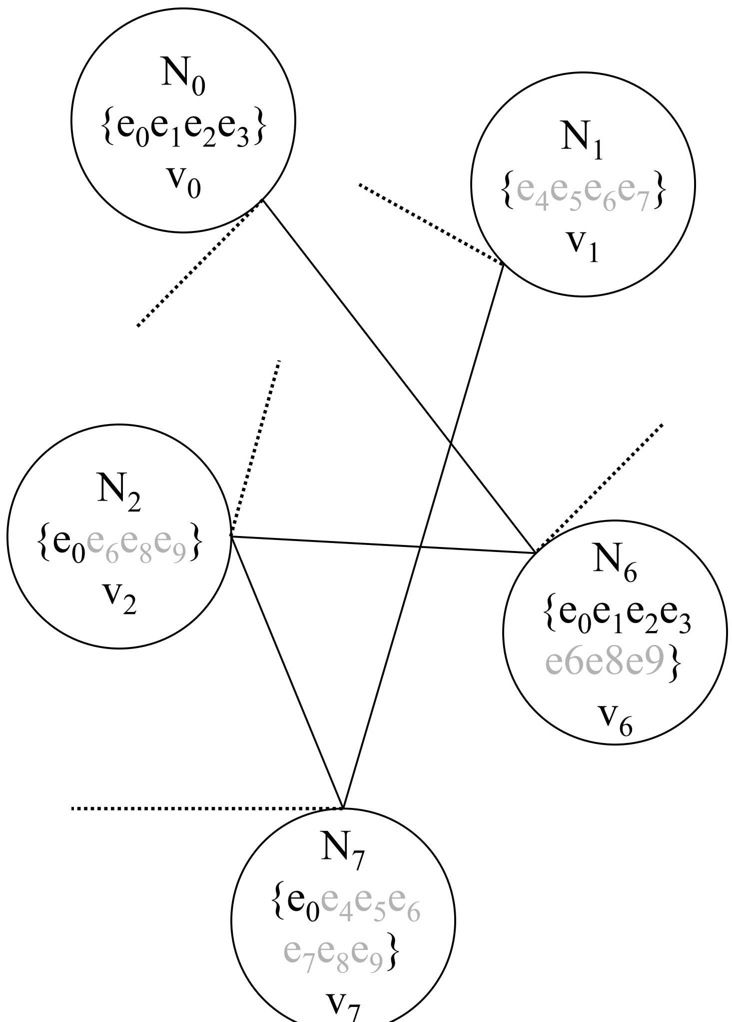


$\cap$ EOI = $\{e_0e_1e_2e_3\}$

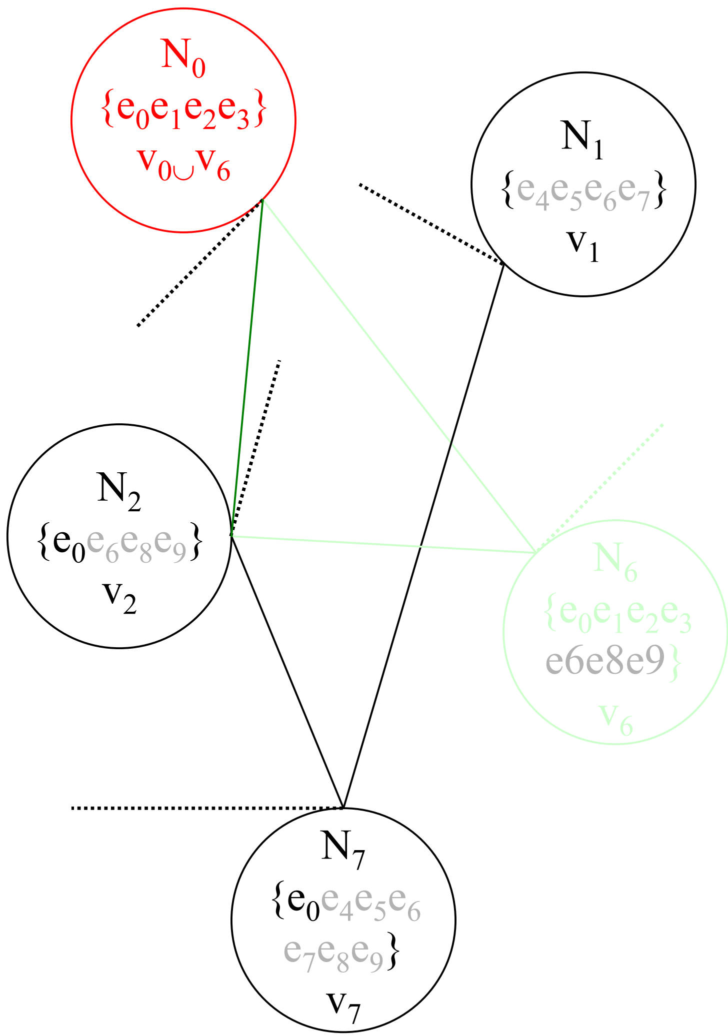
1) Intersect EOIs with entities in AG

Converting AG to EAG Example:

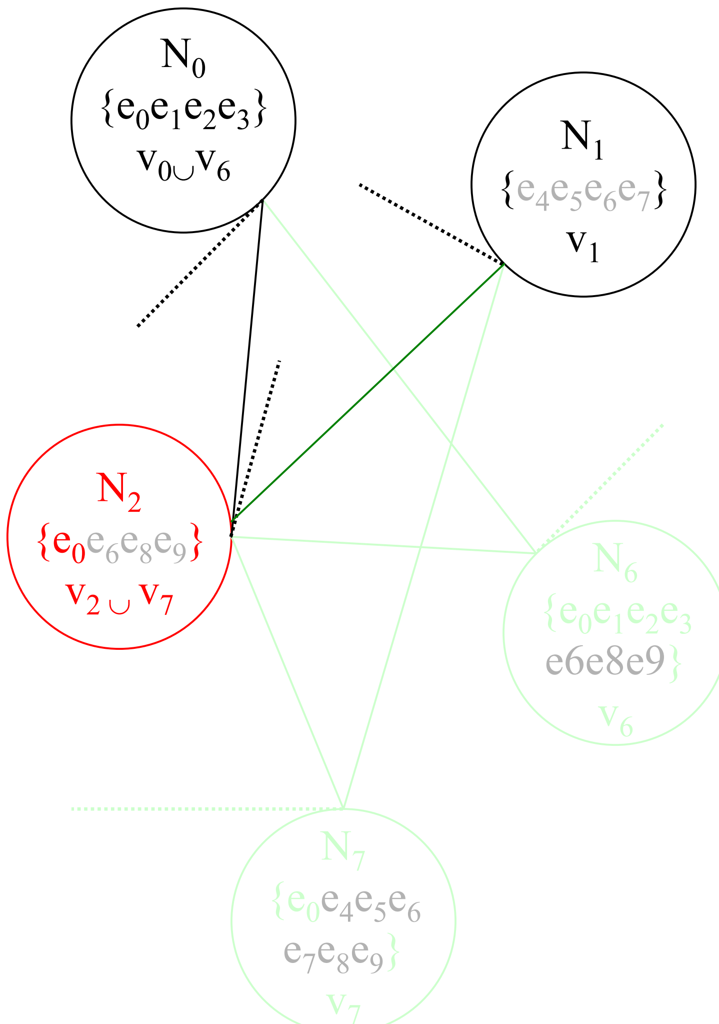
2) merge nodes with same observable entities



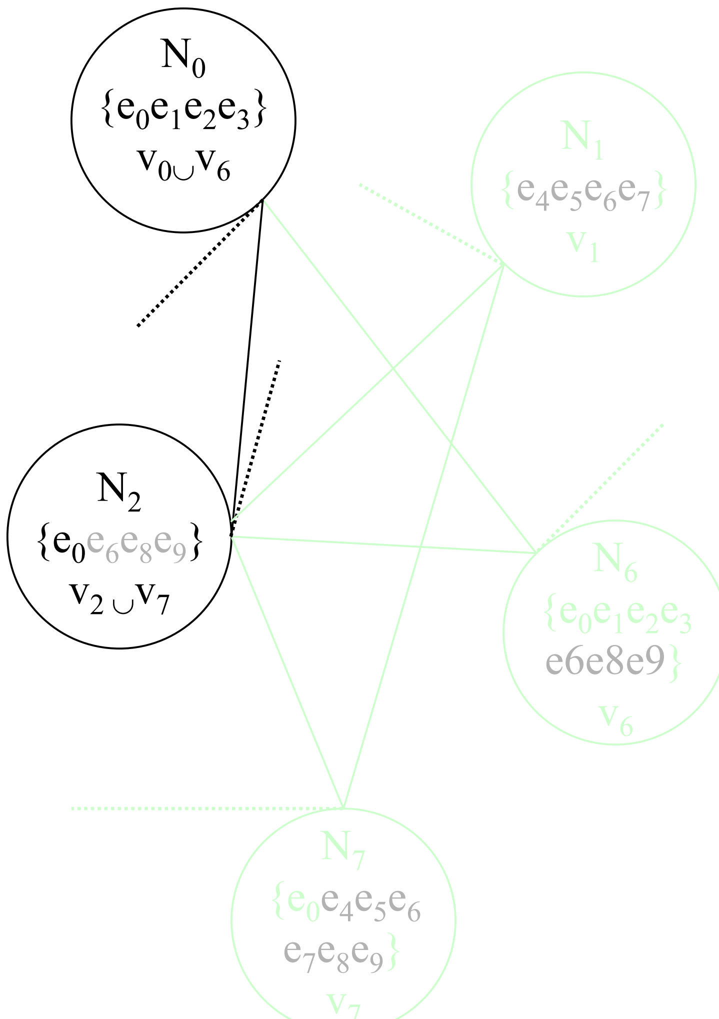
Converting AG to EAG Example: (step 3)



Converting AG to EAG Example:



Converting AG to EAG Example:



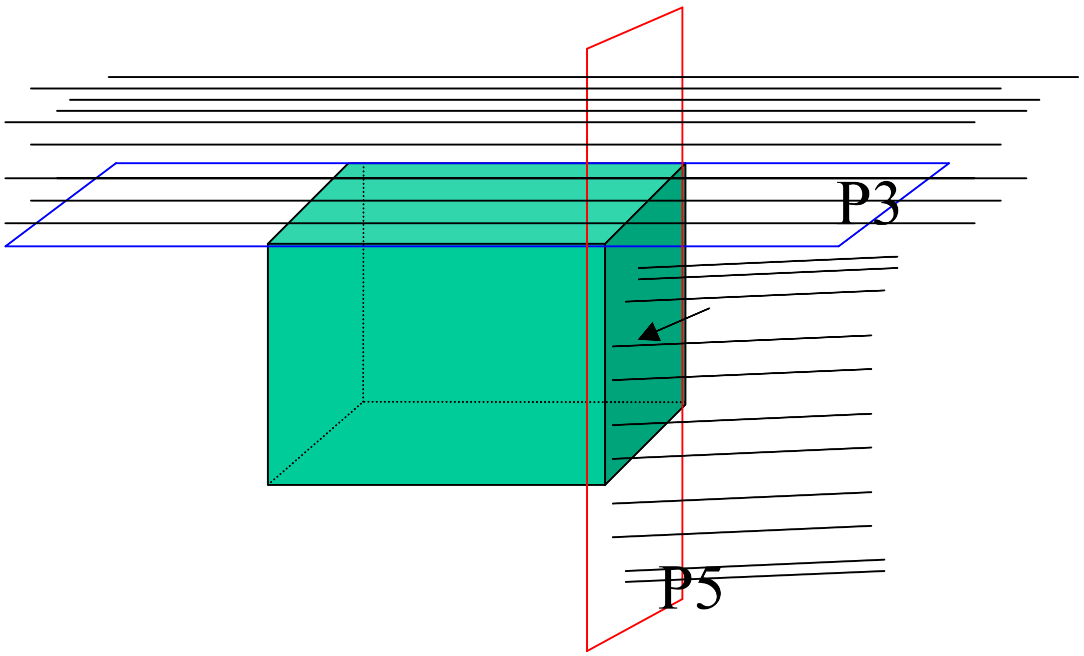
Merging viewing regions

- Issue
- How are two nodes merged?
- How do you calculate the union of two viewing regions?
  - Yang didn't specify a method for this

Union of viewing regions**

- Observations:
  - 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

Example: concave viewing space



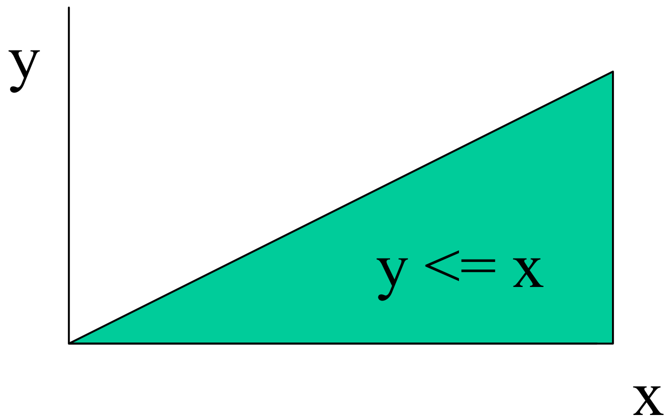
How can we identify such cases?

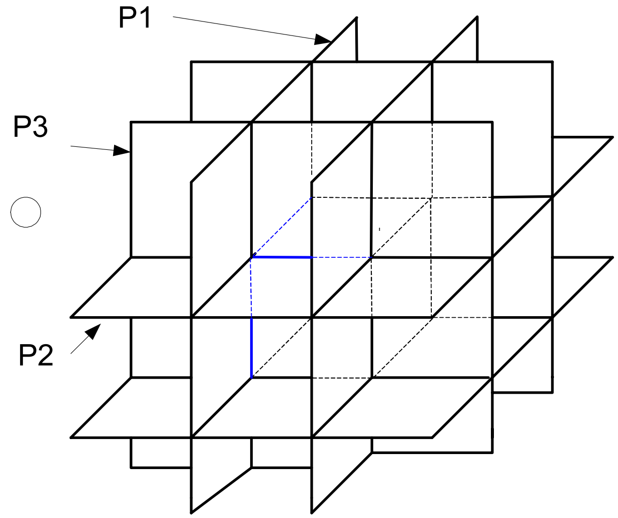
Determining convexity of union of viewing regions

- From: Bemporad A, Fukuda K, Torrisi F. D, *Convexity recognition of the union of polyhedra*, Computational Geometry Theory and Applications, 2001

Determining convexity of union of polyhedra

- Definitions:
- Convex H-Polyhedra
  - Intersection of a finite set of halfspaces of the Euclidean space R^d

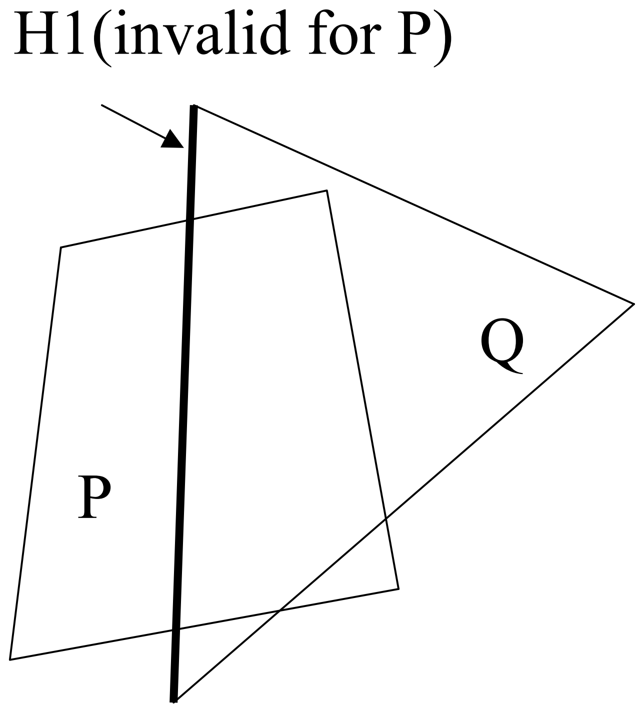


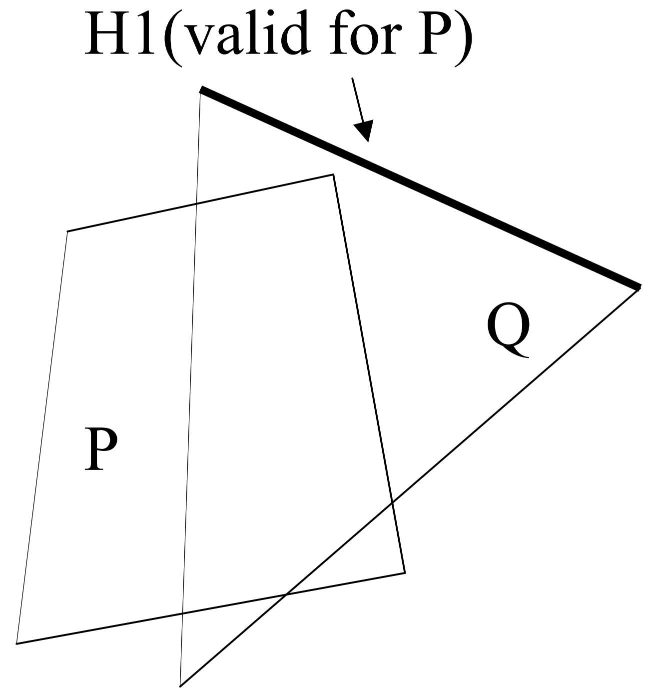


Determining convexity of union of polyhedra Cont.

- Definitions:
- Valid inequality
  - Let P be a convex polyhedron in R^d. An inequality $a^T x <= b$ is called valid for P if it is satisfied by all points in P
  - Conversely, an inequality $a^T x <= b$ is called invalid for P if there exist a point on the other side of the inequality that is in P

Example
valid/invalid inequalities





Identifying valid/invalid inequalities

- Recall,
  - An inequality $a^T x <= b$ is called invalid for P if there exist a point on the other side of the inequality that is in P
- Feasibility Problem
    - Let (A,b) be the inequalities that define P
    - Let (c,t) be the inequality to be tested
  - Max F(x)
  - Subject to: Ax <= b, c > t
  - If there is a feasible point for this problem, the inequality is invalid

Determining convexity of union of polyhedra

- Definitions: Envelope of two polyhedra P and Q
- Let P and Q be (possibly unbounded) H-Polyhedra

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

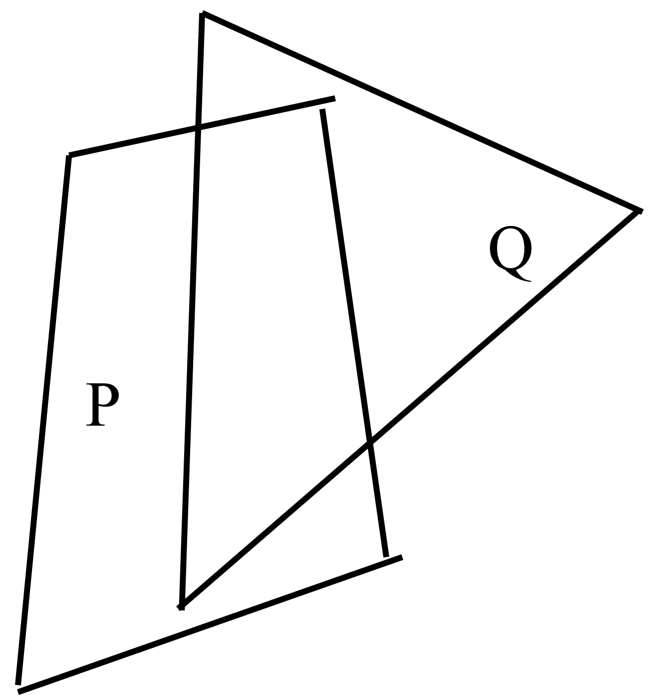
$$Q = \{x \in \Re^d : Bx \le \beta\}.$$

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

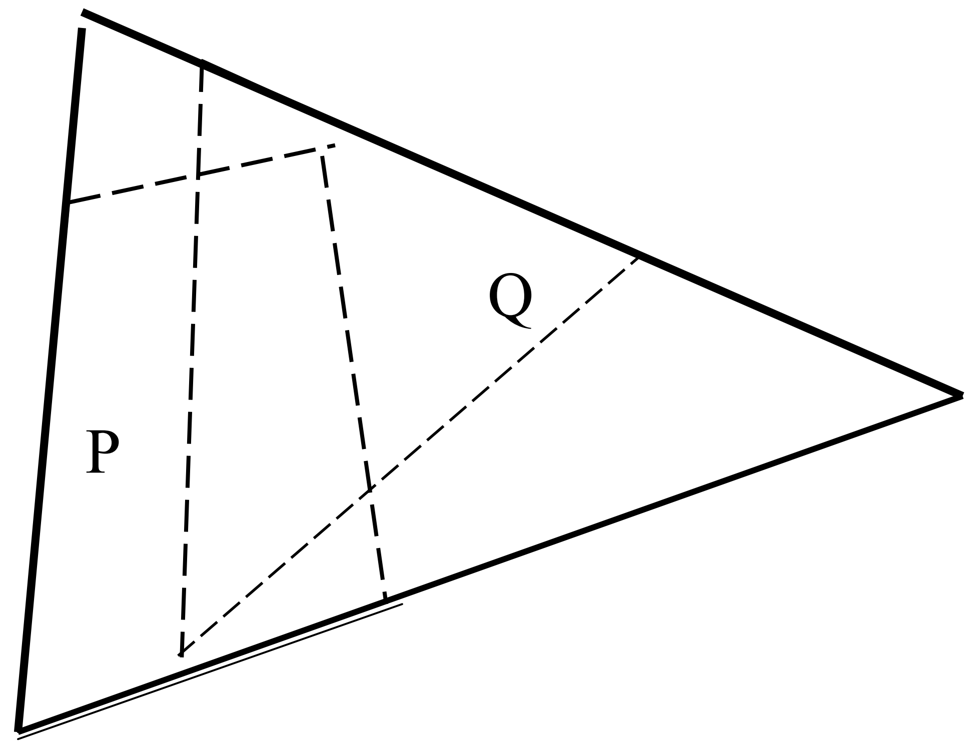
where $\bar{A}x \le \bar{\alpha}$ ($\bar{B}x \le \bar{\beta}$) is the subsystem of $Ax \le \alpha$ ($\bar{B}x \le \bar{\beta}$)

obtained by removing all the inequalities not valid for the other polyhedron Q (P).

Example: env(P,Q) original**



Example: env(P,Q)**



Invalid inequalities ---------------

Env(P,Q) ———

Constructing the env(P,Q)

Let Hp, Hq be the set of hyperplanes that define P, Q respectively

Let C be the set of hyperplanes that define the env(P,Q)

Let Ab be the set of hyperplanes of Q that are not valid for P

Let Bb be the set of hyperplanes of P that are not valid for Q

For each inequality c in Q

    if is_valid(Hp, c) then C = [C; c] else Ab = [Ab; c]

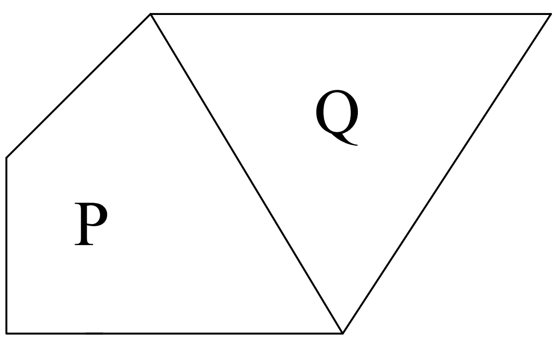
For each inequality c in P

    if is_valid(Hq, c) then C = [C; c] else Bb = [Bb; c]

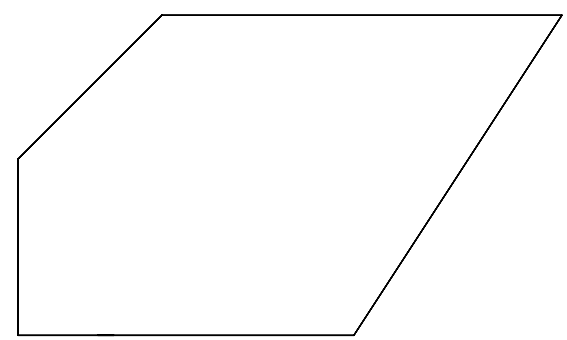
Determining convexity of union of polyhedra**

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

Union(P,Q)

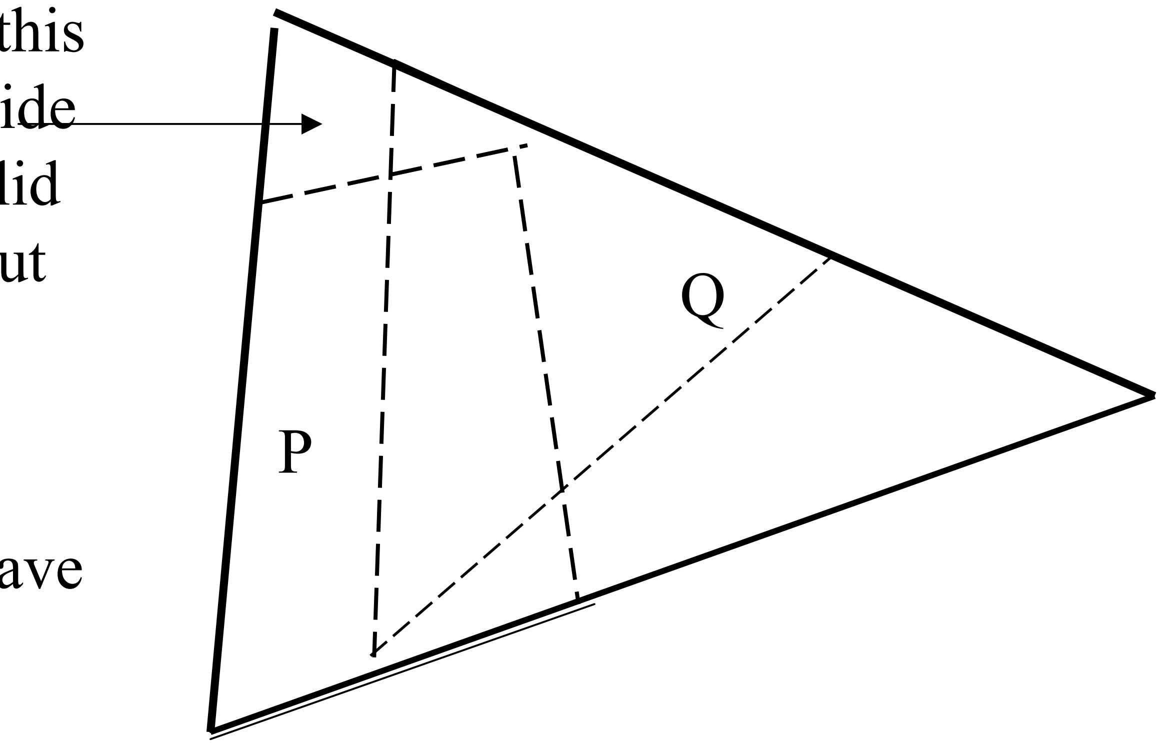


Env(P,Q)



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



Invalid inequalities

Env(P,Q)

Determining convexity of union of viewing polyhedra

- Construct env(P,Q)

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

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

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

    E*= max (x)

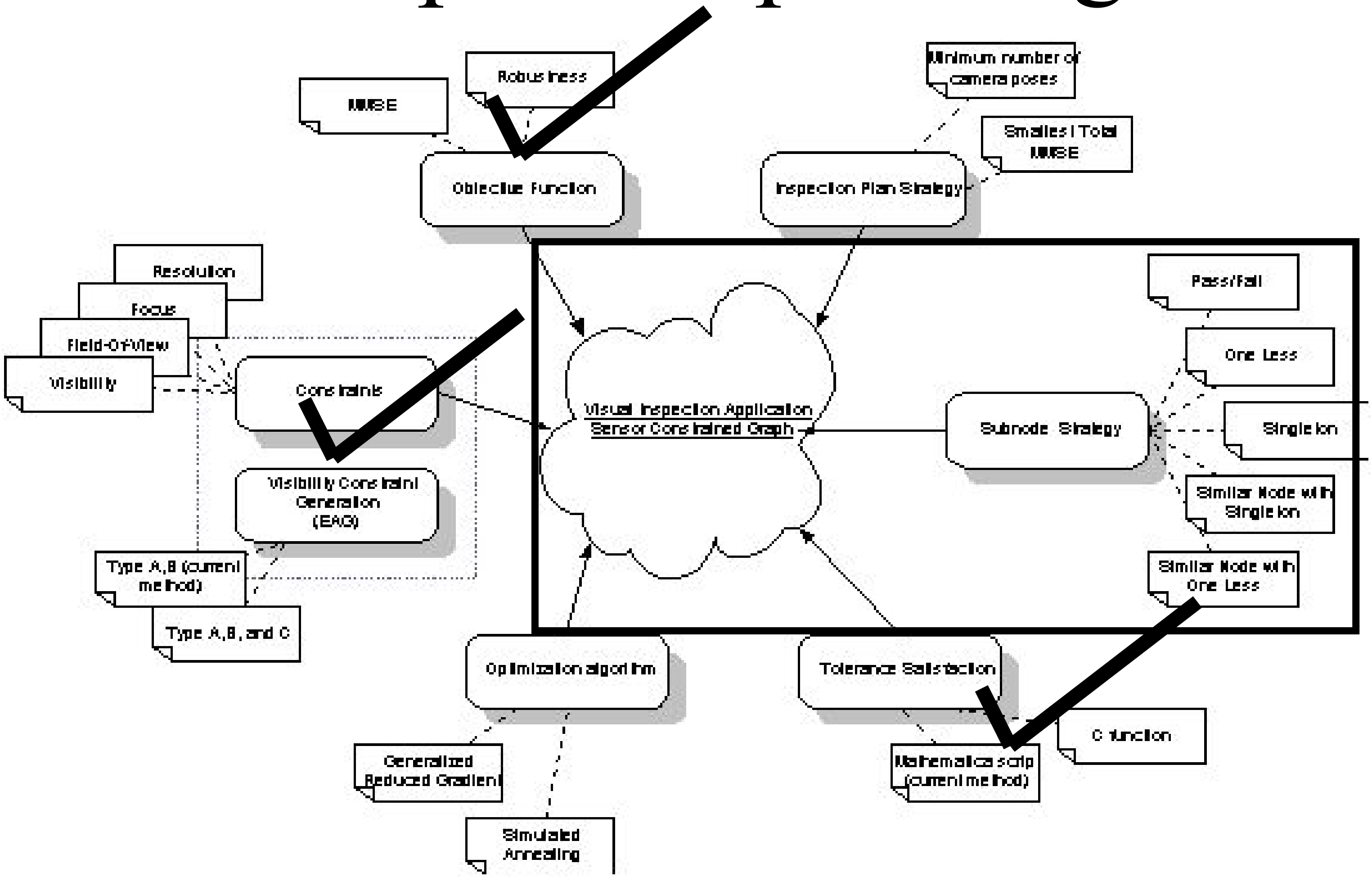
    Subject To: $\overline{A}_i x \geq \overline{\alpha}_i + \varepsilon$, $\overline{B}_j x \geq \overline{\beta}_j + \varepsilon$, $Cx \leq \gamma$

    If feasible return nonconvex

Endfor

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

Inspection planning



Sensor Constraint Graph

- Basic structure used to represent the optimization problems
  - Definition
  - Construction
  - Expansion
  - Contraction
  - Processing using SCG
- The output of this processing is a set SLIST of optimal pose candidates

Sensor Constraint Graph

- SCG node: 4 tuple (E,O,G,I)
  - E: set of desired geometric entities to be observed
  - O: objective function (MSE)
  - 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

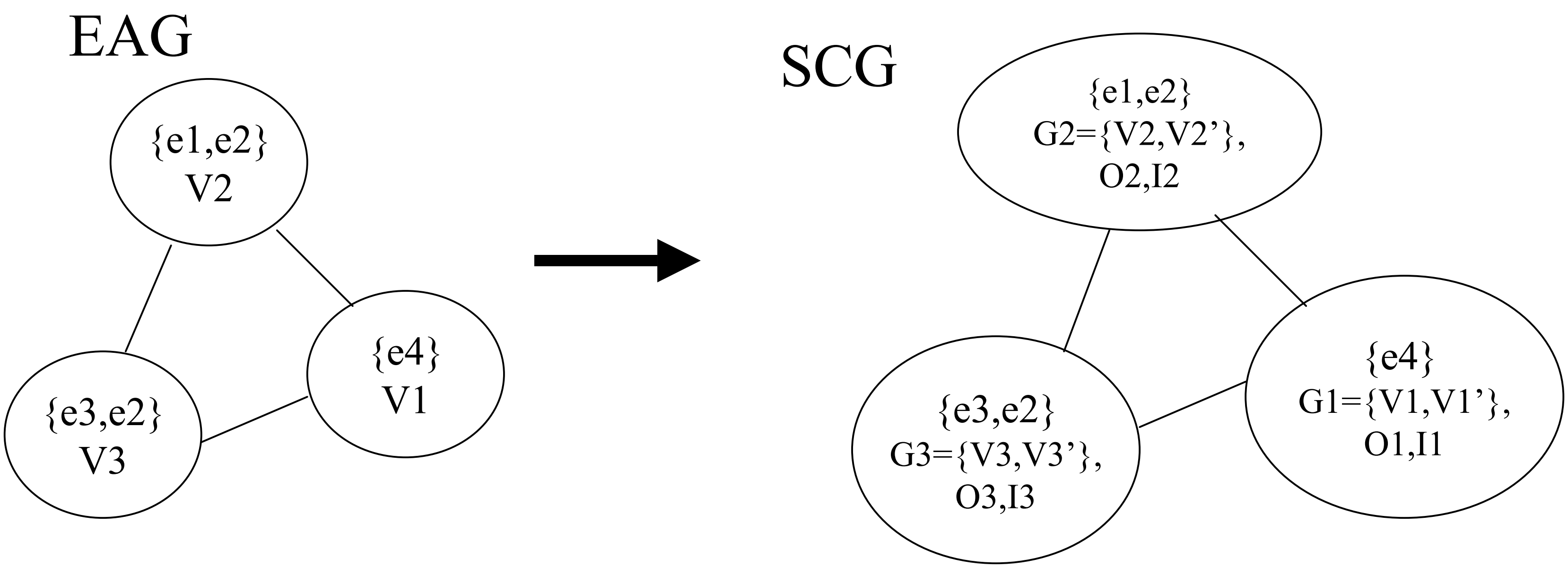
SCG operations

- Three operations:
  - Construction
  - Expansion
  - Contraction

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 arbitrary initial camera pose I
- Link all nodes with solid arcs

Example: SCG construction



Passing Optimizations

- Stored in SLIST
- Node of SLIST defined as S={F,E,O}
  - F camera pose
  - O objective function
  - E set of entities
- SLIST = $\cup$S

Similar Settings

- Nodes that have identical entity sets
- Are combined into new setting S_o such that
  - E_o is set of entities
  - O_o is objective function
  - F_o is the camera pose that results in smallest MSE
- Similar settings are replaced by S_o in the SLIST

Failing Optimizations

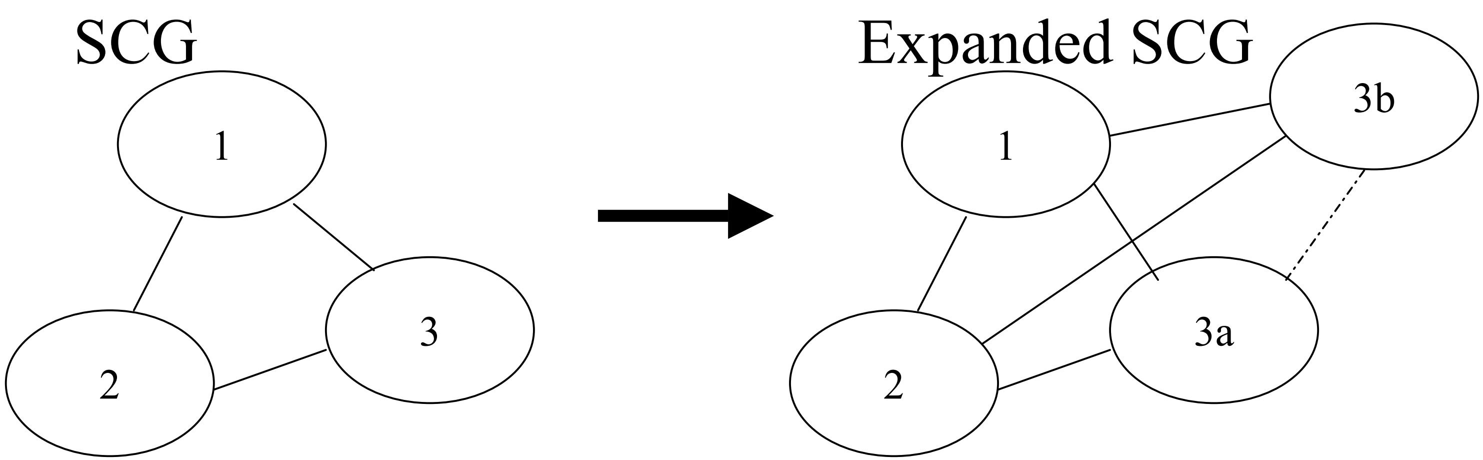
- There exist a set of passing entities and failing entities after an optimization
- Resolved using expansion and contraction operations

SCG expansion

- Creates subnodes
  - Desired entity set is a subset of original node
  - MSE 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



Subnode strategies

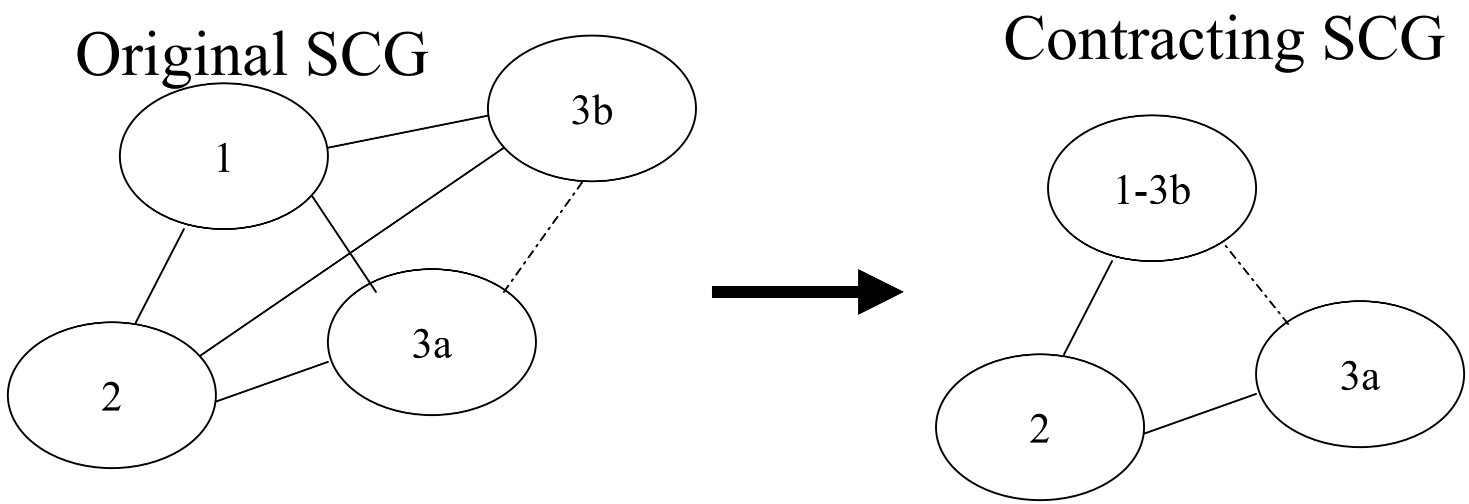
- Two types of failing optimizations
  - passing entities and failing entities in the set
  - only failing entities in the set
- Five strategies to handle these cases
  - Strategy 1: Pass/Fail
  - Strategy 2: One less
  - Strategy 3: Singleton
  - Strategy 4: Similar node with one less
  - Strategy 5: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

SCG contraction

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

SCG contraction

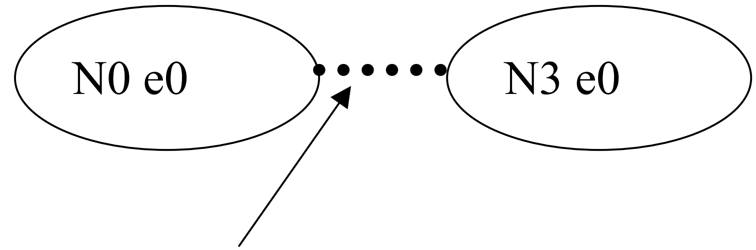
- Neighboring nodes keep same relation with respect to supernode
- Example: contracting node 1 and 3b



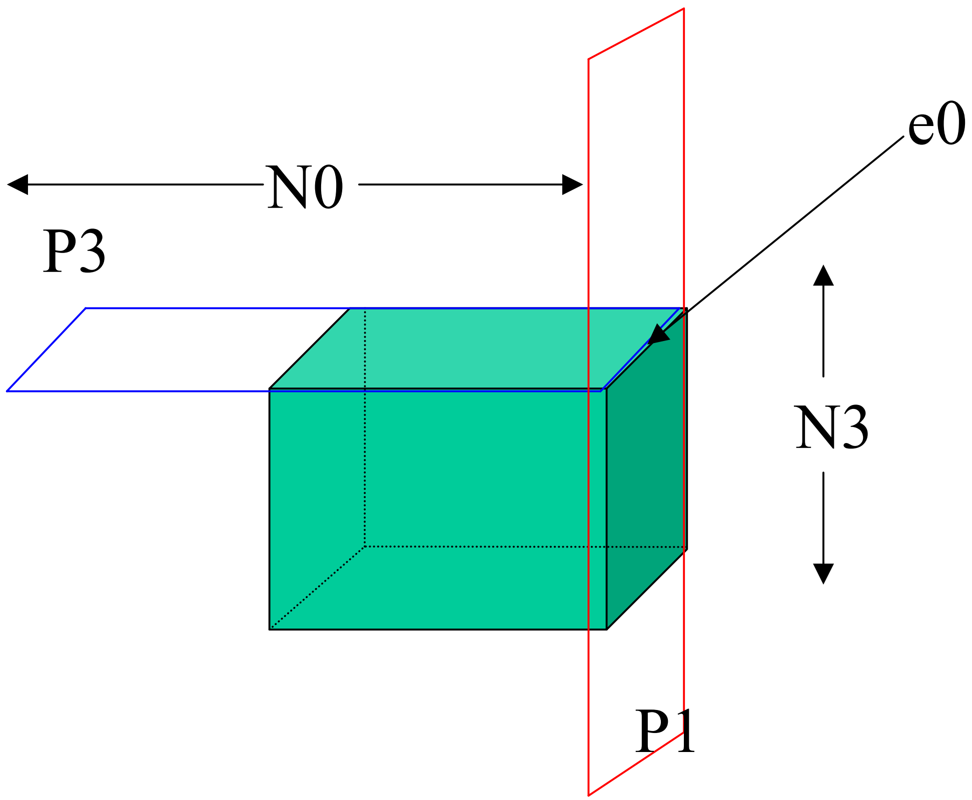
New concepts

- Definition
  - Mergeable nodes: two nodes are mergeable if their viewing regions are mergeable
  - Mergeable viewing regions: two viewing regions are mergeable if their union is a convex viewing region

Conservative contraction



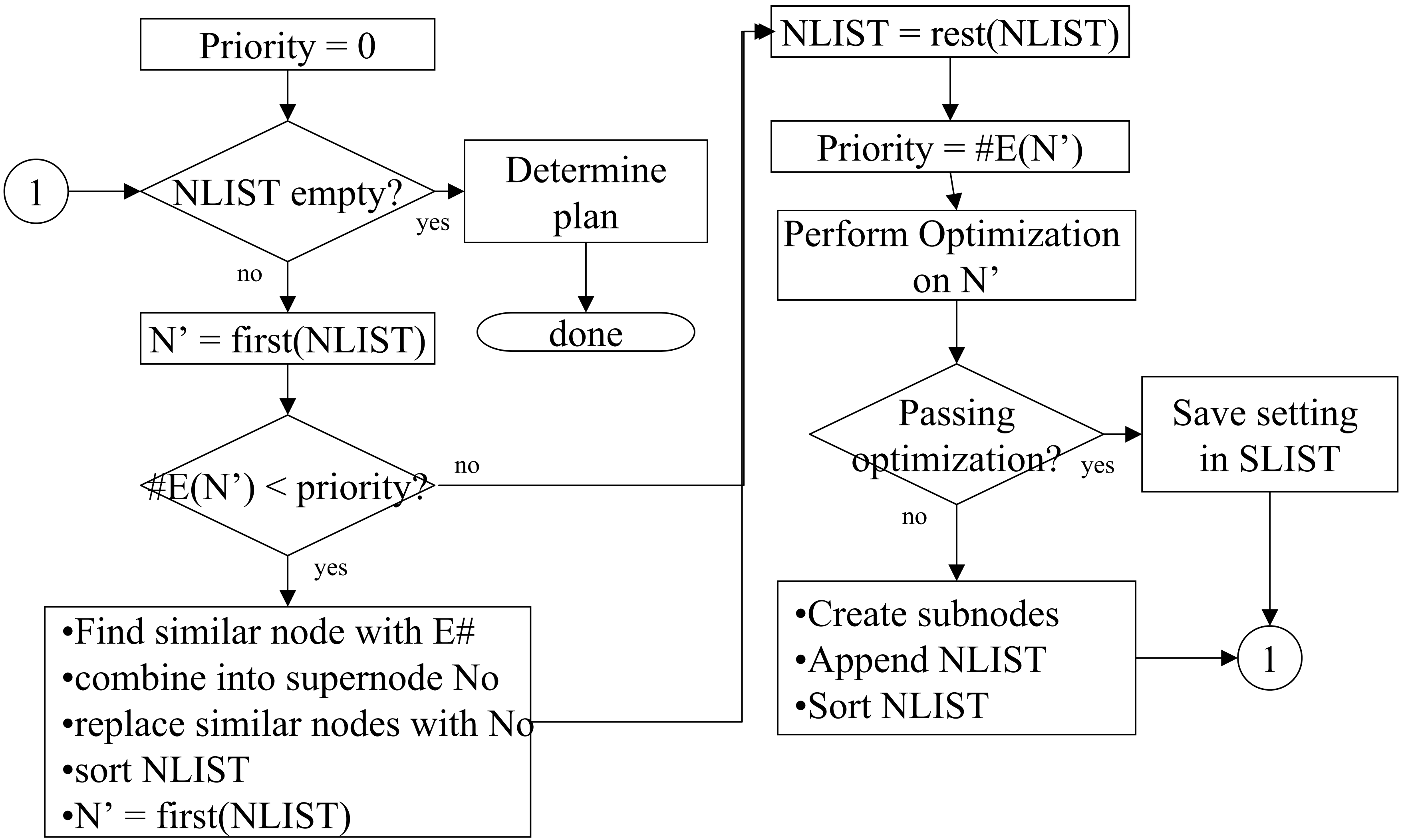
Join nodes with a curly arc to identify unmergeable nodes



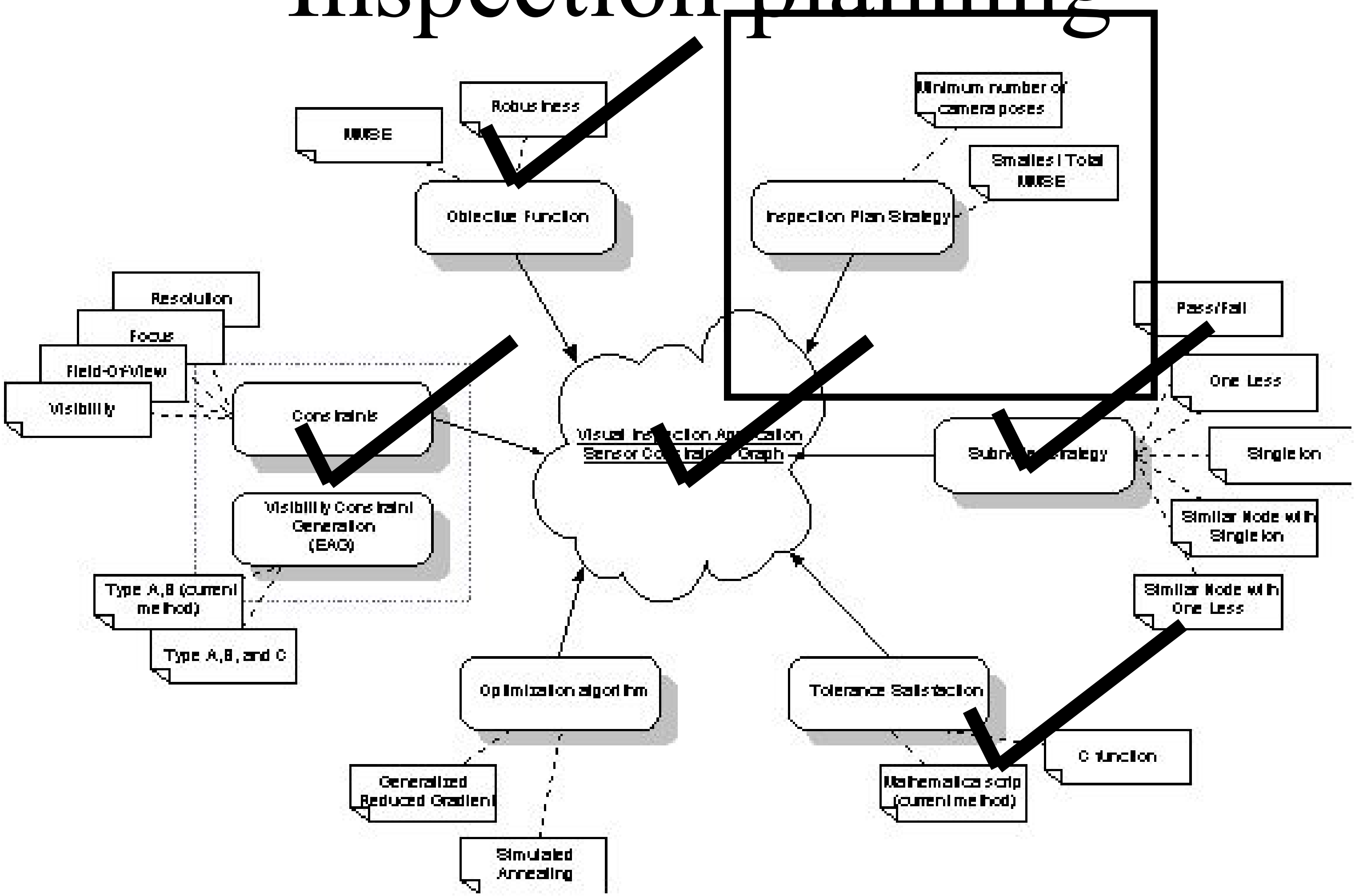
Prioritizing nodes

- Prioritize nodes toward greater cardinality in entity sets to reduce number of optimizations
- Unprocessed nodes are included in priority queue known as NLIST

Prioritizing nodes



Inspection planning



Determining final camera poses

- An integer program is used to the determine the final set S

Integer Programming

<u>Input</u>

Let

p be the number of camera poses in SLIST

x_i = 1 if i^{th} camera pose S_i is part of inspection plan for i=1 to p

a_{ij} = 1 if entity E_j is in setting S_I

n be the total number of desired entities

F(**x**) be a optimality function

<u>Output</u>

An optimal inspection plan **x** that indicates the set of cameras that minimize the objective function F(**x**)

<u>Algorithm</u>

$$Minimize : F(x)$$

$$S.T.$$

$$Ax \geq 1$$

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

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

Outline

- Introduction and Motivation
- Existing Related Works
- Problem Formulation and Statement
- Solution Approach
- Details
- **Comparison with Related Work**
- Plan for Work

Comparison with related work

- This work vs. Crosby, Yang
  - based on VIPER, SCG, and EAG concepts
  - completes formulation of EAG by defining algorithm to determine the union of viewing volumes
  - uses different error model since 3D measurements can’t be done using a single image
  - uses stereo camera setup to perform 3D measurements

Comparison with related work

- This work vs. Mason, Olague
  - Follows the philosophy of Mason and Olague of using more than one camera to perform accurate measurements
  - Visibility calculations use viewing sphere and search is done using heuristics (Mason) and genetic algorithms (Olague) vs. Aspect Graph and nonlinear programming (This Work)
  - Mason uses predefined networks originally specified by expert photogrammetrists and updates them to satisfy new requirements. Olague generates poses using genetic algorithms. Both systems handle more than 2 cameras. Usually 4 or more.
  - This work only deals with stereo

Comparison with related work

- This work vs. Tarabanis
  - Uses the same sensor constraints and adds Room Size, Incidence Angle constraints from Mason’s work
- EAG, AG, Union of Viewing Volumes
  - Stewman’s Aspect Graph code
  - modified Yang’s Entity-Based Aspect Graph algorithm and incorporated Bemporad’s Union of Polyhedra algorithm

- only 2 cams

Comparison with related work

- Limits of current implementation
  - No illumination planning is done
  - Only two cameras are used
  - Only edges are measured
  - Only polyhedral objects are used
  - Optimal solution depends on initial feasible pose
  - It was shown that searching the viewing space is a NP complete problem [Mason] (using heuristics is the sensible way of achieving a suboptimal solution) <-- we don't do this
  - Using only two image points to estimate 3D position doesn't give a statistically significant number of observations, therefore the LSE estimates may be inaccurate [Mason] <-- That's why photogrammetrists use at least 4 cameras!

Outline

- Introduction and Motivation
- Existing Related Works
- Problem Formulation and Statement
- Solution Approach
- Details
- Comparison with Related Work
- **Plan for Work**

Plan of work

- Incorporate new error model into the framework
- Run simulations to verify it
- Run real experiments