Automatically Generating Virtual Guided Tours

Tsai-Yen Li, Jyh-Ming Lien,
Shih-Yen Chiu, and Tzong-Hann Yu

Computer Science Department
National Chengchi University
Taipei, Taiwan, R.O.C.
Email: {li, s8415, s8433, s8410}@cs.nccu.edu.tw

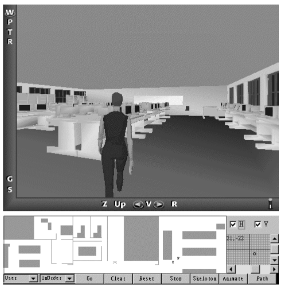
CA' 99 **May, 1999**

Outline of the Talk

- **Motivation**
- **Problem descriptions and related work**
  - customized tour path planning
  - real-time humanoid simulation
  - intelligent camera motion planning
- **Proposed approaches**
  - decoupled planning approach
  - greedy approach for optimal sequences
  - constrained kinematics approach for human motion
- **System architecture and Implementation**
- **Experimental results**
- **Conclusion and future work**

Motivations

- **Networked virtual environment problems :**
  - frame rate is low for complex scenes
  - user control is too low-level
- **Proposal: <u>an auto-navigation system</u>**
  - specifying locations of interests by clicking on a 2D layout map
  - system generates guided tours using motion planners
- **Featuring: <u>interactive</u>**
  - **tour path planning**
  - **humanoid simulation**
  - **camera motion planning**



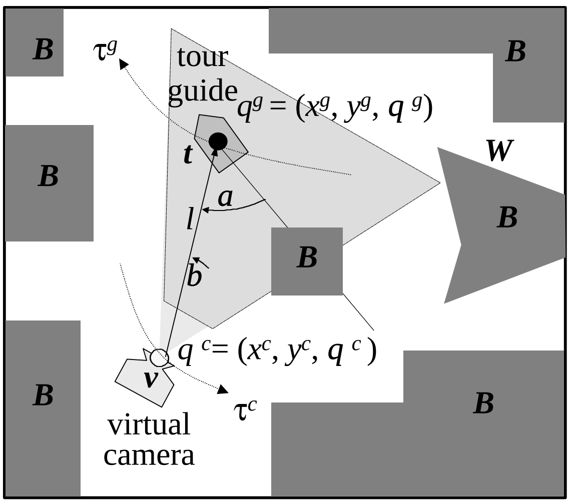
Problem Descriptions

- **<u>Tour path planning</u>:** given an environment description and points of interests, to find a good (if not optimal) tour path that is
  - passing through all these points
  - suitable for a human tour guide to follow
- **<u>Humanoid motion simulation</u>:** given a sequence of footsteps, to generate human walking gaits in real time.
- **<u>Camera motion planning</u>:** given an environment description and a tour path, to find a legal camera motion that is
  - collision free from the obstacles
  - keeping the guide in sight all the time

Related Work

- **Tour path planning:**
  - Piano Mover's Problem
    - [Reif 79], [Latombe 91], [Barraquand 91], etc.
  - Travelling Saleperson's Problem
    - [Cormen 94], etc.
- **Humanoid simulation:**
  - Motion generation (w/ or w/o dynamics)
    - [Girard 85], [Sims 88], [Badler 97], [Huang99], etc.
  - Modifying captured motions
    - [Unuma 95], [Witkin 95], [Rose 96], [Hodgins 97], etc.
- **Camera motion planning:**
  - Film directing with Cinematographic idioms
    - [Drucker 95], [He 96], etc.
  - Intelligent observer problem
    - [Drucker 94], [Becker 95], **[Lavalle 97]**, etc.

Problem Formulation: Configuration Spaces



Tour guide configuration:
$q^g = (x^g, y^g, q^g)$

Camera configuration:
$q^c = (x^c, y^c, q^c)$

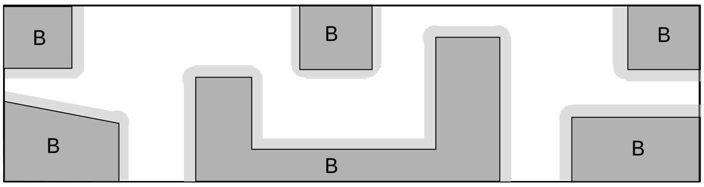
Composite space:
$(x^g, y^g, q^g, x^c, y^c, q^c)$

Tour guide's C-space:
$C(x^g, y^g, q^g)$

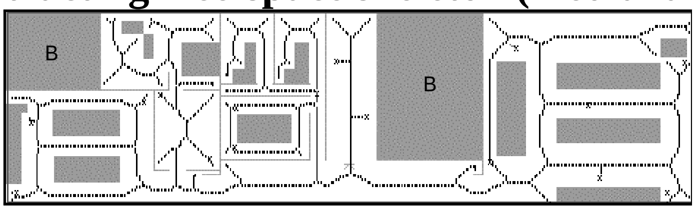
Camera's CT-space:
$CT(t, x^c, y^c, q^c)$

Capturing Free-Space Structure for Tour Path Planning

- **Simplifying tour guide to an enclosing circle.**
  - growing workspace obstacles to form C-space.

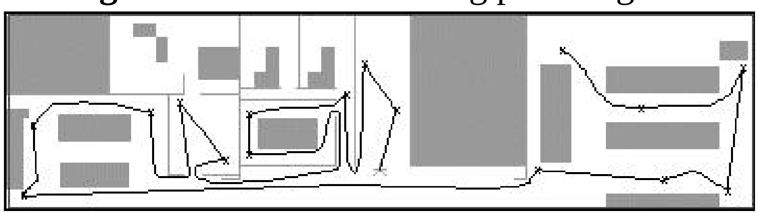


- **Extracting free-space skeleton (medial axis).**

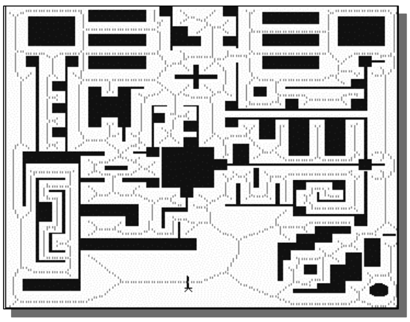


Tour Path Planning: Optimal Sequence Problem

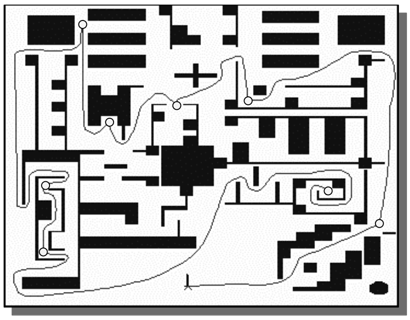
- **<u>Greedy approach</u> for finding the tour path and a near-optimal traversing order**
  - doing breadth-first search on skeleton to find the nearest unvisited location of interest.
  - starting another search from this new location until all locations are visited.
- **Smoothing a path:** replacing sharp turns with Bezier curves
- **Tour guide orientation:** facing path tangent



Tour Path Planning: Another Tour Path Example



Free-space skeleton



Typical planned tour path

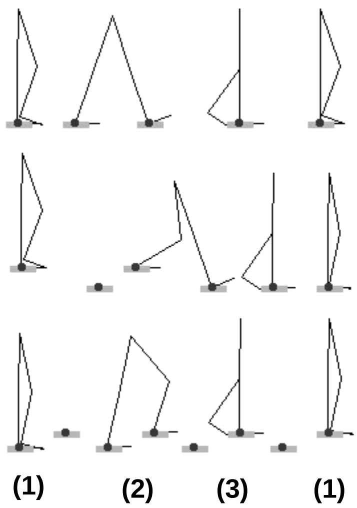
Kinematics Approach for Real-time Humanoid Simulation

Given: footprint locations
To Find: human lower body motion

(a) Leveled Ground

(b) Down Hill

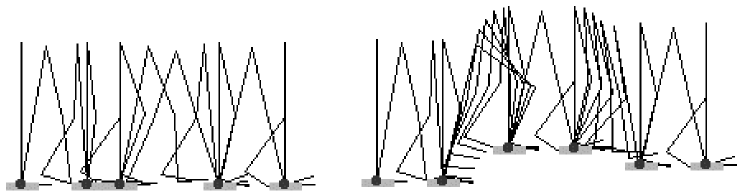
(c) Up Hill



(1) kick off
(2) touch ground
(3) regain balance

Key Frame Interpolation for Real-time Humanoid Simulation

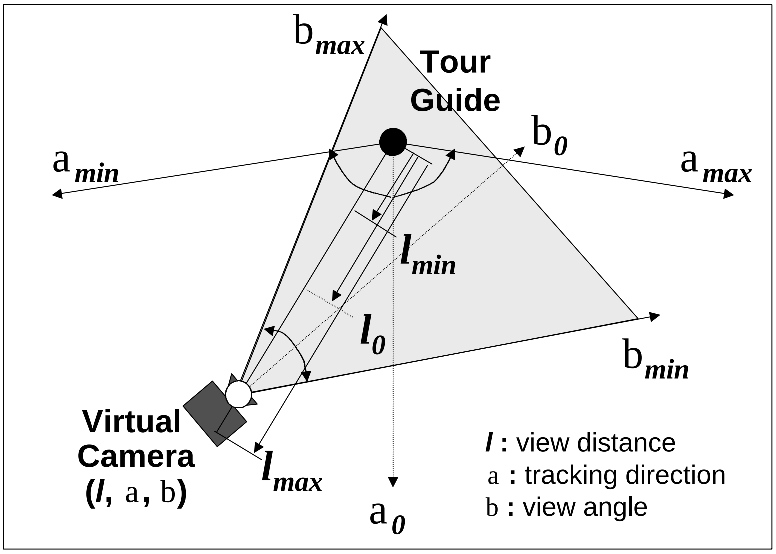
- **Interpolation Principles:**
  - Leg on the ground: joint space interpolation
  - Leg in the air: Cartesian space interpolation



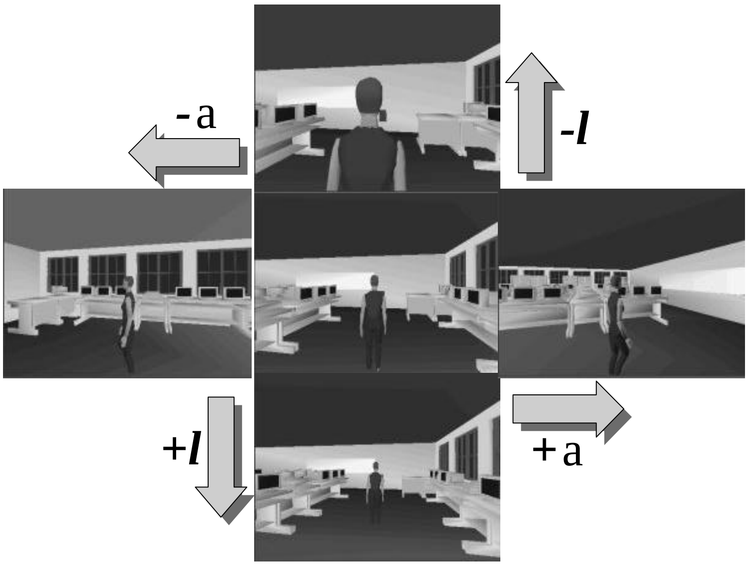
on leveled ground   on stairs

Key frame interpolation for variable step sizes

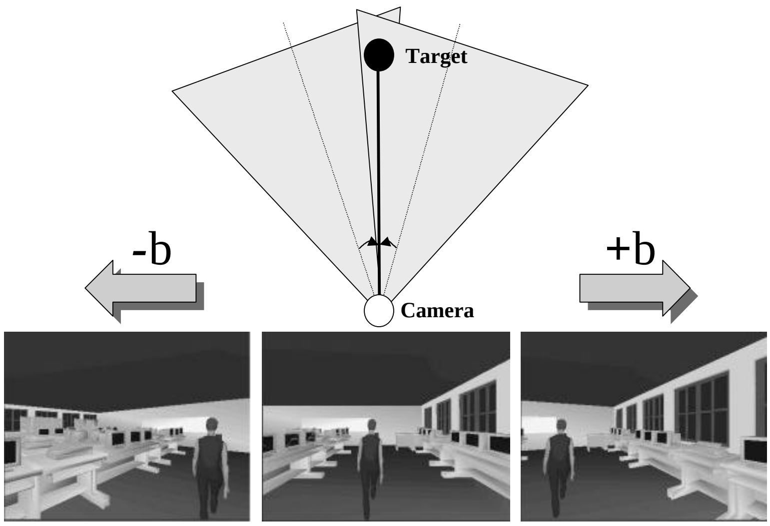
Problem Formulation: Parameterization for Camera Motion



View Model: View Distance (*l*) and Tracking Direction (a)

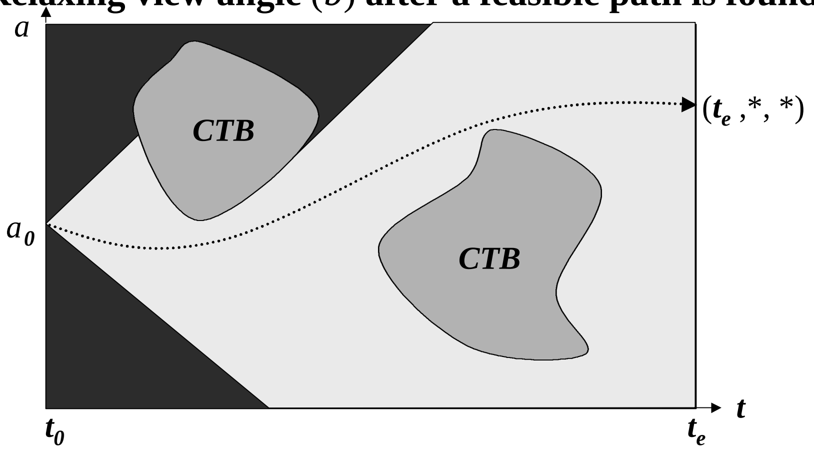


View Model : View Angle (b)



Search Space for Camera Motion Planning

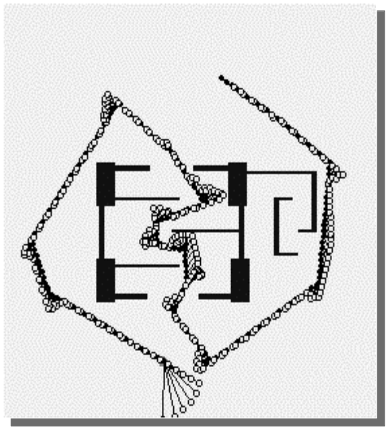
- **Equivalent space:** $CT\ (t, x^c, y^c, q^c)$ => $CT'\ (t, l, a, b)$
- **Simplification:** fixing view angle (b) => $CT''\ (t, l, a)$
- **Relaxing view angle** (b) **after a feasible path is found.**



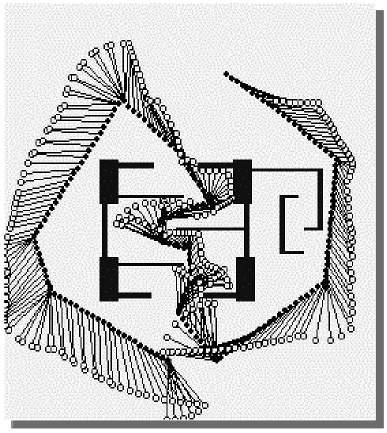
Search Criteria for Camera Planning

- **Planning time (*t*):** highest priority
  - return the first found path
- **View Distance (*l*):** subjective criterion
  - percentage of the human figure in the rendered image
- **Tracking Direction (a):** subjective criterion
  - a range centered behind the target
- **Overall Movement (*d*):** subjective criterion
  - avoid motion sickness and speed up graphics
- **View Angle (b):** lazy movement in postprocessing
  - avoid frequent rotation/scene changes

An Example of Camera Motion

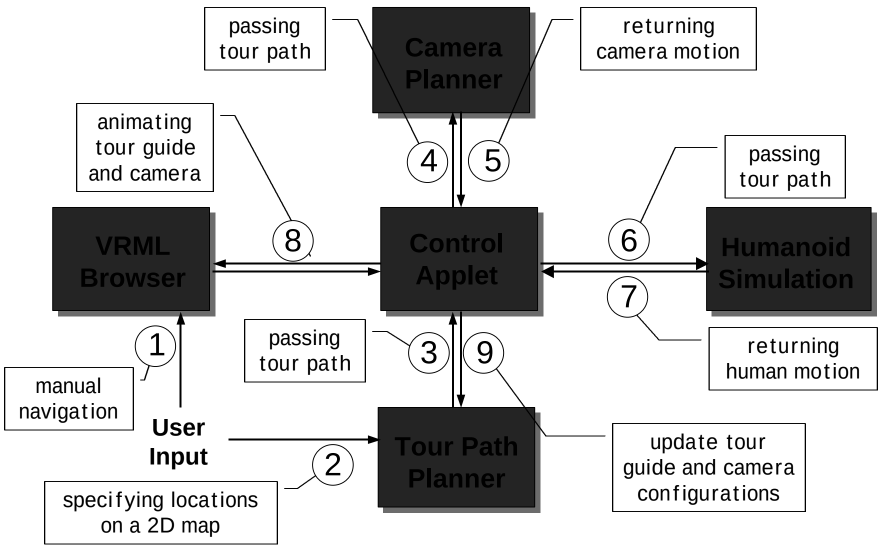


Prefer good tracking direction (a)



Prefer good view distance (*l*)

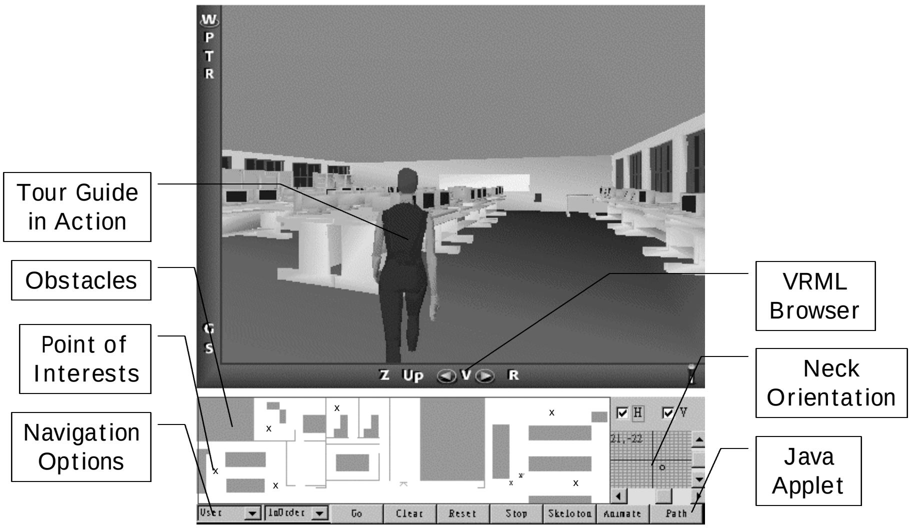
System Architecture: a Typical Data Flow Diagram



Implementations and Experiments

- **All modules except for the VRML browser are implemented as Java applets.**
- **Applets communication:**
  - **object scripting model in WWW browser.**
- **Control applet <-->VRML browser:**
  - **External Authoring Interface (EAI)**
- **Building geometric models: ~1.3MB (0.5-4 fps)**
  - 2D layout map was created separately.
- **Tour guide model:**
  - conforming to the VRML humanoid specification
- **Experimental platform:**
  - **Planning times measured on a Pentium II 450 PC**

Experimental Result: Graphical User Interface

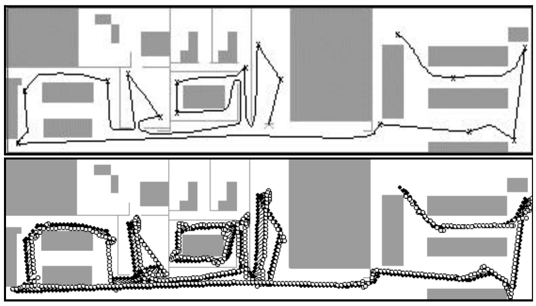


Experimental Results: Planning Efficiency

- **Tour path search space:**
  - **560 x 150 grid, rotational increment: 10 degrees**
- **Camera path search space:**
  - **resolution of (*t*, *l*, a): 1112 x 50 x 100**
- **Planning time for tour path and camera path:**
  - **tour path: 0.19 sec. camera path: 0.38 sec.**

Tour Path:

Camera Path:



Conclusion and Future Work

- **Proposing an auto-navigation system capable of**
  - **generating <u>customized tour paths</u>**
  - **generating <u>humanoid walking animation</u> in real-time**
  - **generating <u>intelligent camera motions</u>**

- **Future extensions:**
  - **richer contents during the tour**
  - **finding optimal traversing sequence**
  - **incorporating more user interactions during the tour**
  - **applications in virtual factories, virtual malls, etc.**

Q & A

Best-First Planning Algorithm

```
procedure BFP {
  mark all the configurations in CT¢¢ as unvisited;
  INSERT(qi, OPEN); mark qi as visited;
  SUCCESS ¬ false;
  while (!EMPTY(OPEN) and !SUCCESS) {
    q ¬ FIRST(OPEN);
    for (every q¢ Î NEIGHBOR(q)) {
        mark q¢ visited;
        if (LEGAL(q¢ )) {
            PARENT(q¢) ¬ q;
            INSERT(q¢, OPEN);
        }
        if (GOAL(q¢)) SUCCESS ¬ true;
    }
  }
  if (SUCCESS)
    return the path by tracking back to qi
}
```

Planning Criteria: Cost Functions

$$f(t, f, l, dir) = w_1 * f_1(t) + w_2 * f_2(f) + w_3 * f_3(l) + w_4 * f_4(f, l, dir)$$

$f_1(t) = t_e - t$, **cost function for the <u>time difference</u> to the ending slices**

$f_2(f) = | f - f_0 |$, **cost function for the <u>tracking direction</u>**

$f_3(l) = | l - l_0 |$, **cost function for the <u>view distance</u>**

$f_4(f, l, dir) = dist(p(f, l, 0)-p(f, l, dir))$, **cost function for the <u>Euclidean distance</u> moved from its parent configuration,**

w_i*:** normalized weights (except for ***w_1) for individual cost functions,

***t*:** current time,
***t_e*:** is the ending time

***f*:** current tracking direction,
***f_0*:** is a neutral tracking direction

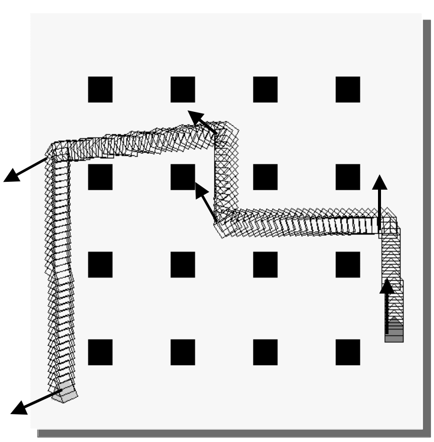
***l*:** distance between the viewpoint and the target,
***l_0*:** is a neutral view distance

***dir*:** an integer indicating the direction where the current configuration was created,

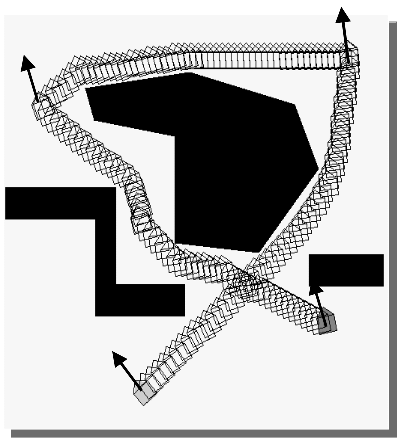
***p*:** returns the previous position of the viewpoint for the given approaching direction,

***dist*:** returns the distance between two positions.

Experimental Examples: Target Path



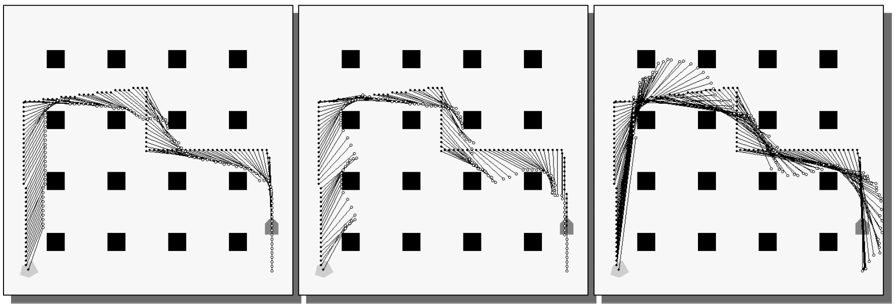
Example 1: 257 steps



Example 2: 515 steps

<u>Generated by a Holonomic Path Planner</u>

Experimental Results: An Example



Parameter Set #1

w_2 (a)	= 100.0
w_3	= 0.0
w_4	= 0.0
planning time	= 0.56 sec

Parameter Set #2

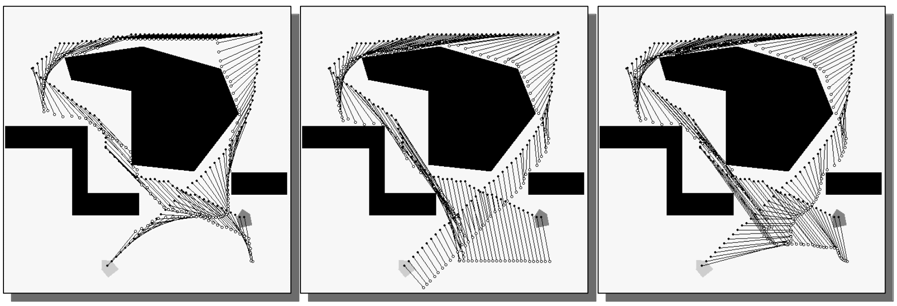
w_2	= 0.0
w_3 (l)	= 100.0
w_4	= 0.0
planning time	= 0.39 sec

Parameter Set #3

w_2	= 0.0
w_3	= 0.0
w_4 ($dist$)	= 100.0
planning time	= 2.59 sec

Camera Tracking Motions

Experimental Results: Another Example



Parameter Set #1

w_2 (a)	= 100.0
w_3	= 0.0
w_4	= 0.0
planning time	= 1.86 sec

Parameter Set #2

w_2	= 0.0
w_3 (l)	= 100.0
w_4	= 0.0
planning time	= 2.07 sec

Parameter Set #3

w_2	= 0.0
w_3	= 0.0
w_4 ($dist$)	= 100.0
planning time	= 5.00 sec

Camera Tracking Motions