

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

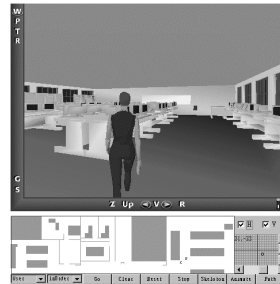
May, 1999

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: an auto-navigation system**
 - specifying locations of interests by clicking on a 2D layout map
 - system generates guided tours using motion planners
- **Featuring: interactive**
 - **tour path planning**
 - **humanoid simulation**
 - **camera motion planning**



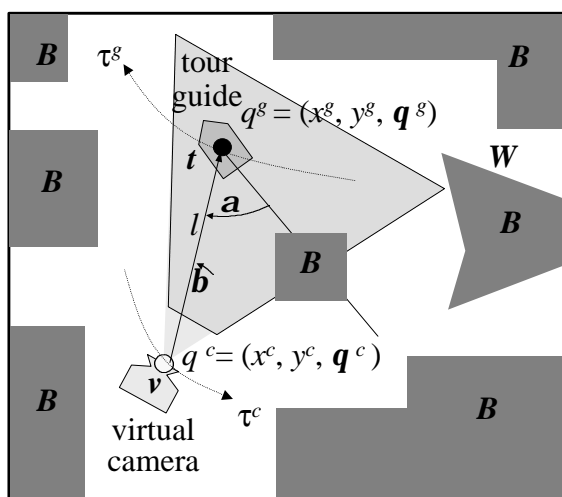
Problem Descriptions

- **Tour path planning**: 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
- **Humanoid motion simulation**: given a sequence of footsteps, to generate human walking gaits in real time.
- **Camera motion planning**: 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 Salesperson'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], [Lavage 97], etc.

Problem Formulation: Configuration Spaces



Tour guide configuration:
 $q^g = (x^g, y^g, \mathbf{q}^g)$

Camera configuration:
 $q^c = (x^c, y^c, \mathbf{q}^c)$

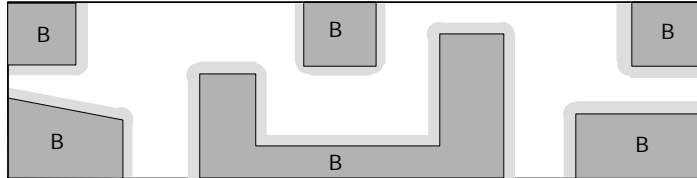
Composite space:
 $(x^g, y^g, \mathbf{q}^g, x^c, y^c, \mathbf{q}^c)$

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

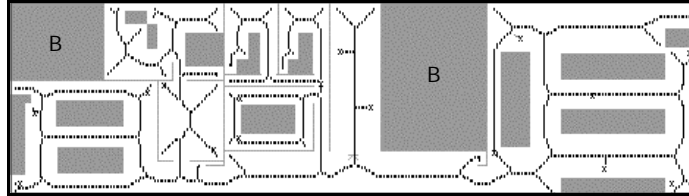
Camera's CT-space:
 $CT(t, x^c, y^c, \mathbf{q}^c)$

Capturing Free-Space Structure for Tour Path Planning

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

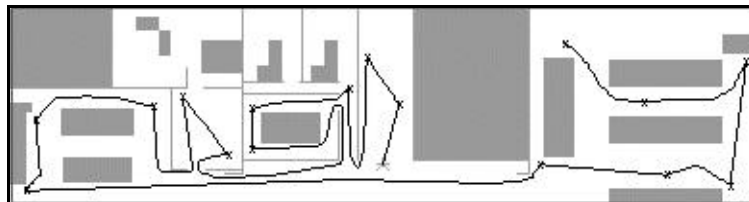


- **Extracting free-space skeleton (medial axis).**

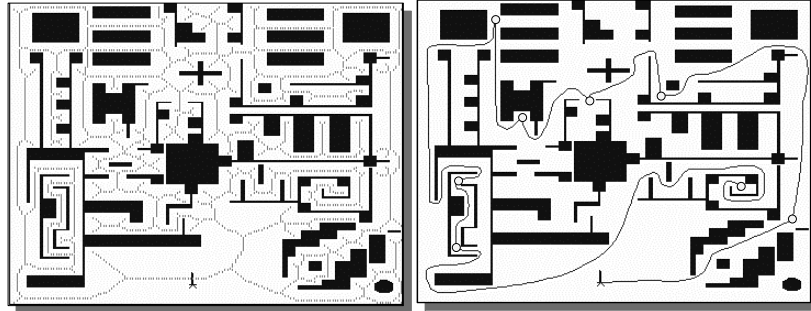


Tour Path Planning: Optimal Sequence Problem

- **Greedy approach 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

(a)
Leveled
Ground

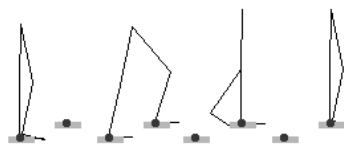


Given: footprint
locations
To Find: human
lower body motion

(b)
Down Hill



(c)
Up Hill



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

(1)

(2)

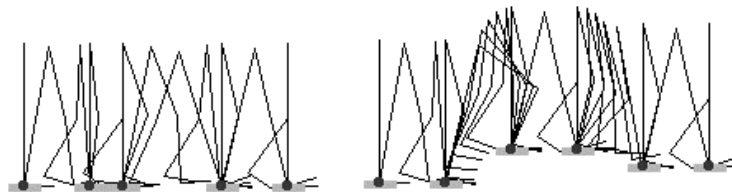
(3)

(1)

Key Frame Interpolation for Real-time Humanoid Simulation

- **Interpolation Principles:**

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

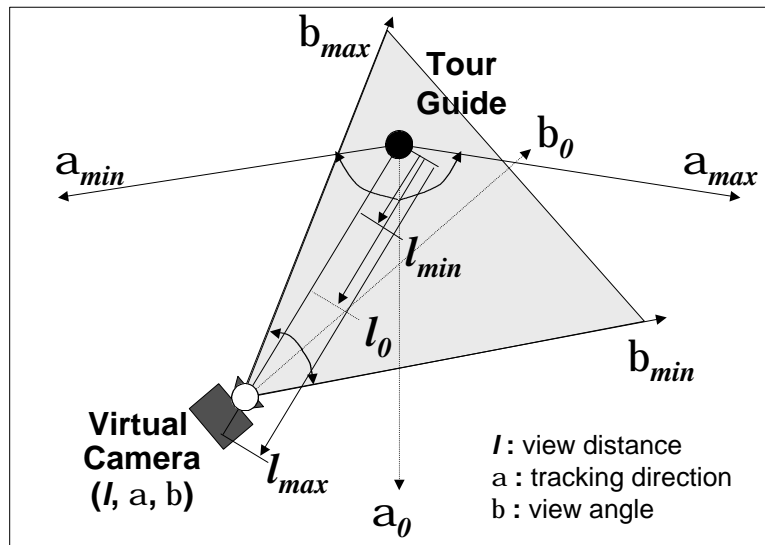


on leveled ground

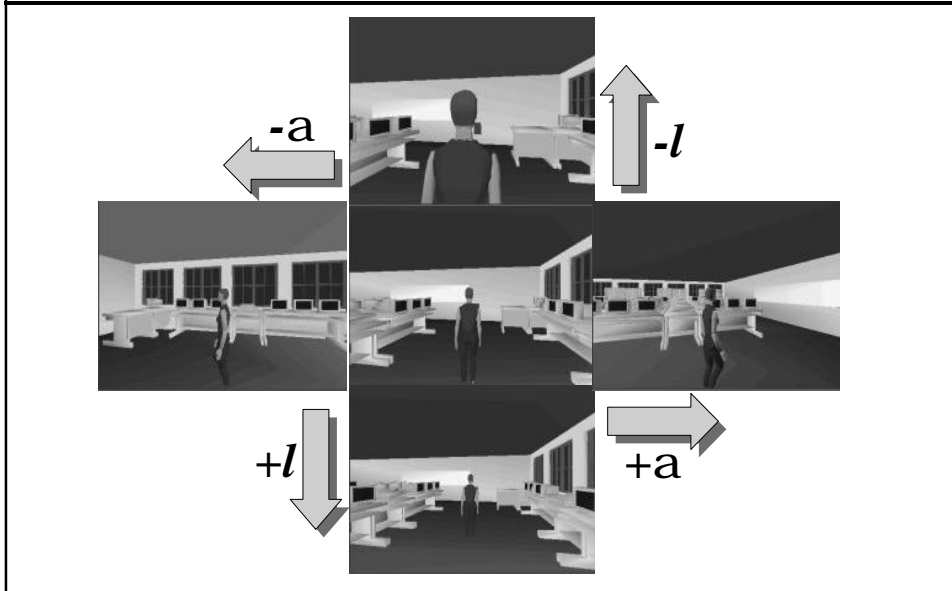
on stairs

Key frame interpolation for variable step sizes

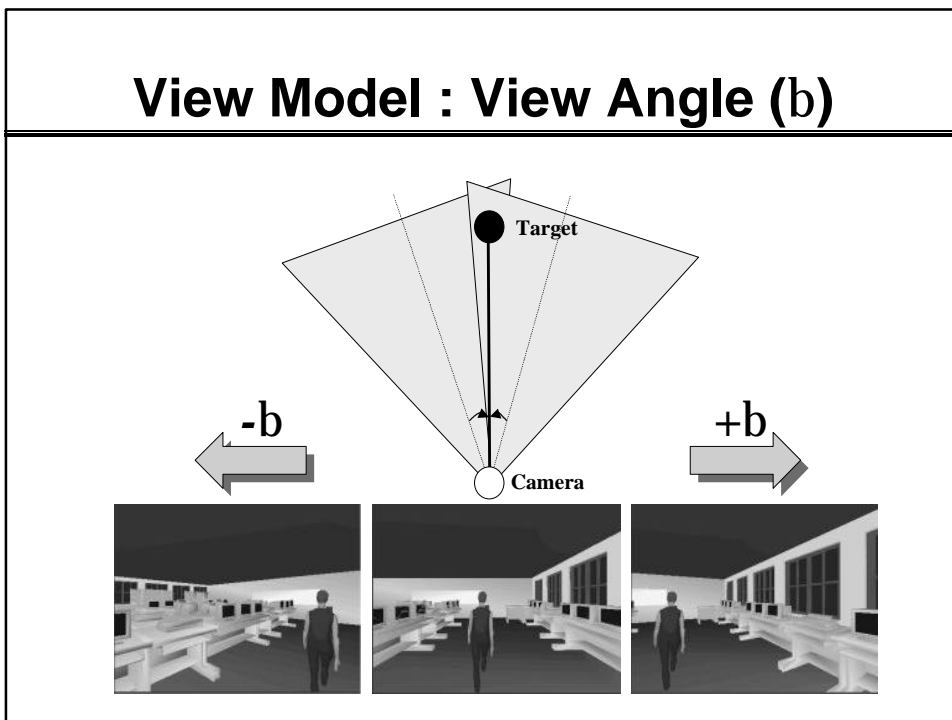
Problem Formulation: Parameterization for Camera Motion



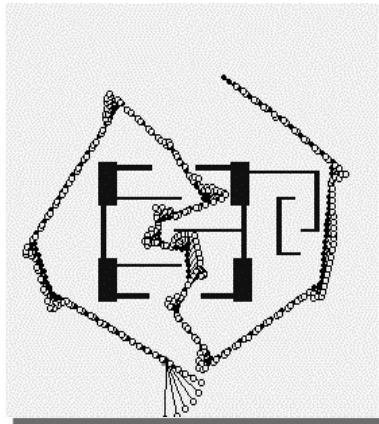
View Model: View Distance (l) and Tracking Direction (a)



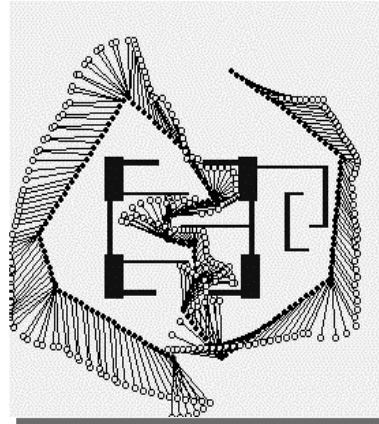
View Model : View Angle (b)



An Example of Camera Motion

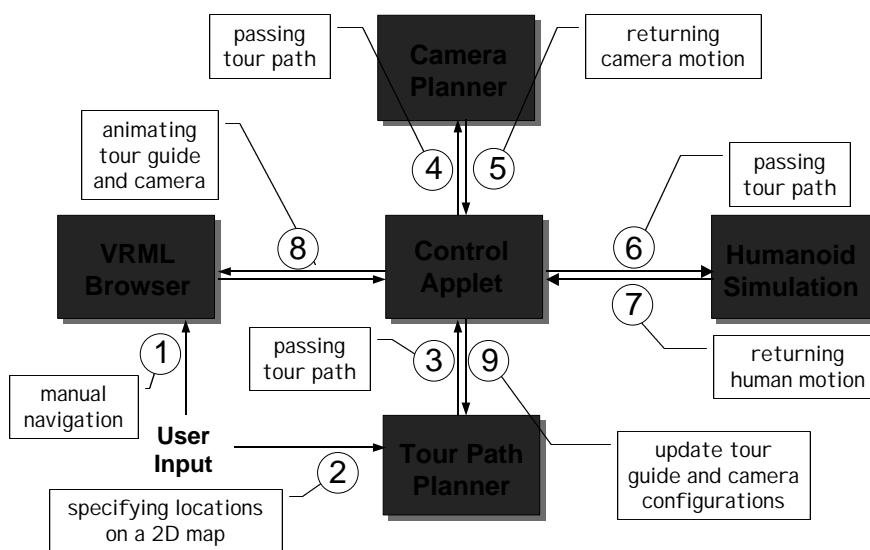


Prefer good tracking direction (a)



Prefer good view distance (l)

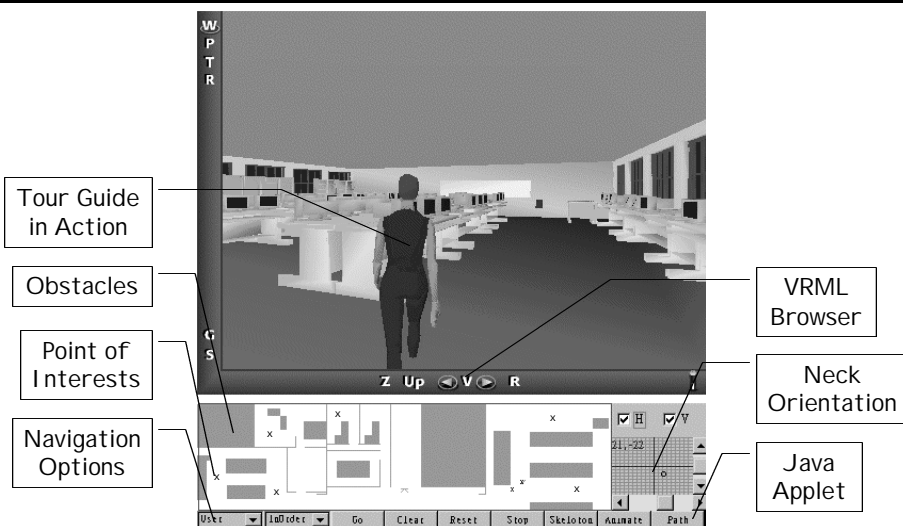
System Architecture: a Typical Data Flow Diagram



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



Q & A

Best-First Planning Algorithm

```
procedure BFP {
  mark all the configurations in  $CT_0$  as unvisited;
  INSERT( $q_i$ , OPEN); mark  $q_i$  as visited;
  SUCCESS ← false;
  while (!EMPTY(OPEN) and !SUCCESS) {
     $q$  ← FIRST(OPEN);
    for (every  $q^c$  ∈ NEIGHBOR( $q$ )) {
      mark  $q^c$  visited;
      if (LEGAL( $q^c$ )) {
        PARENT( $q^c$ ) ←  $q$ ;
        INSERT( $q^c$ , OPEN);
      }
      if (GOAL( $q^c$ )) SUCCESS ← true;
    }
  }
  if (SUCCESS)
    return the path by tracking back to  $q_i$ 
}
```

Planning Criteria: Cost Functions

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

$f_1(t) = t_e - t$, cost function for the time difference to the ending slices

$f_2(\underline{f}) = | \underline{f} - \underline{f}_0 |$, cost function for the tracking direction

$f_3(l) = | l - l_0 |$, cost function for the view distance

$f_4(\underline{f}, l, dir) = dist(p(\underline{f}, l, 0) - p(\underline{f}, l, dir))$, cost function for the Euclidean distance 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

\underline{f} : current tracking direction,

\underline{f}_0 : is a neutral tracking direction

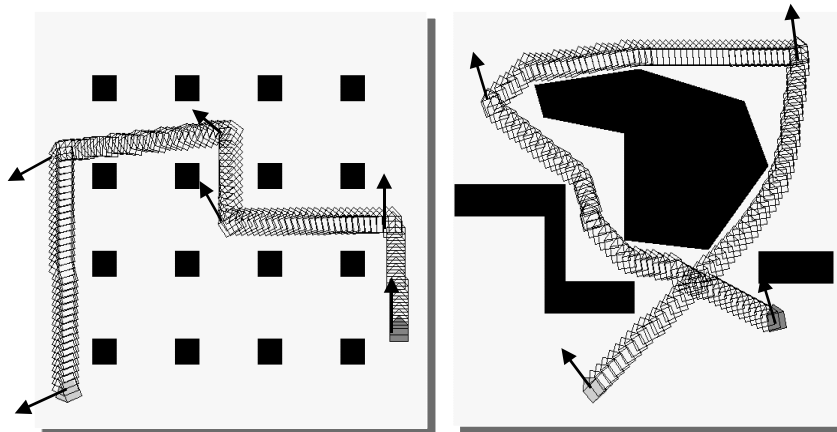
l : distance between the viewpoint and the target, l_0 : is a neutral view distance

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

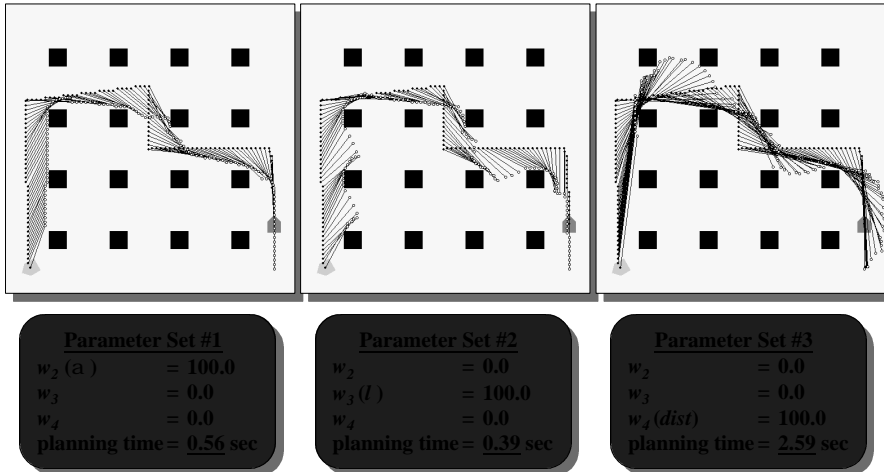


Example 1: 257 steps

Example 2: 515 steps

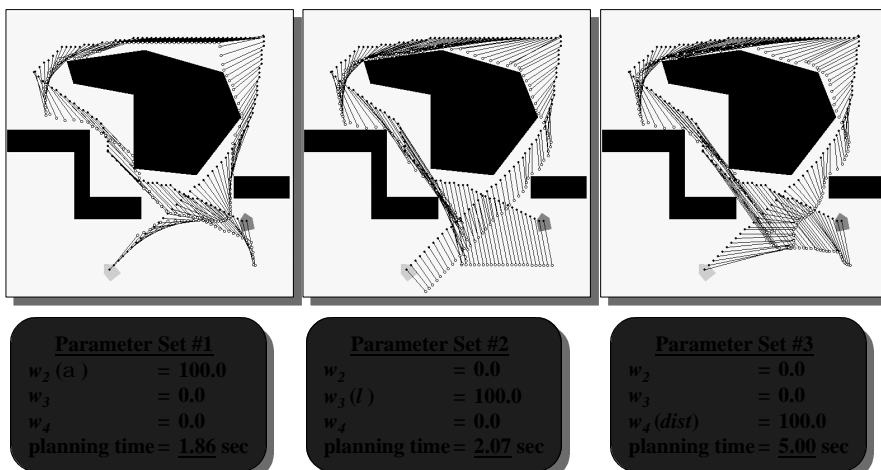
Generated by a Holonomic Path Planner

Experimental Results: An Example



Camera Tracking Motions

Experimental Results: Another Example



Camera Tracking Motions