Today's presentation:

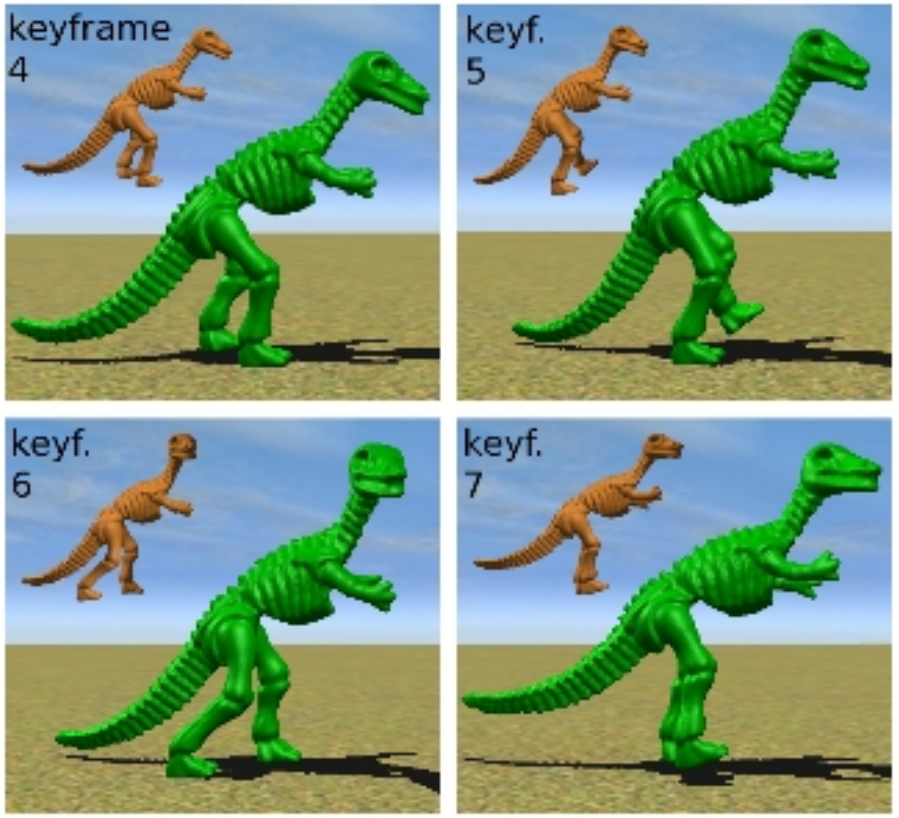
Two totally unrelated topics

- Deformable Object Animation Using Reduced Optimal Control

- The Graph Camera

Deformable Object Animation Using Reduced Optimal Control

Jernej Barbic, Marco da Silva (MIT), and
Jovan Popovic (MIT and Adobe Systems, Inc.)



A fast space-time optimization method for authoring physically-based deformable object simulations that conform to animator-specified keyframes

Control and Auto-magical Physics

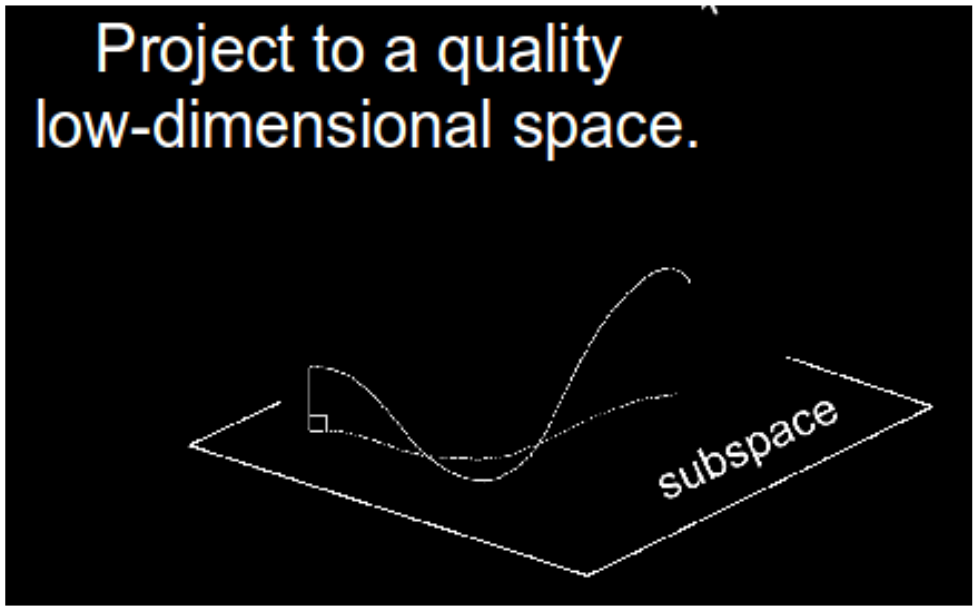
- This paper describes a way to have both!
- Given keyframes for a deformable model
  - Finite Element Method model
  - Mass-Spring system
- Use fake control forces to steer the physically-based model towards the keyframes
- Minimize a weighted sum:
  - The deviations from the keyframes
  - Amount of fake control forces needed

Space-Time Optimization

- They use space-time optimization to minimize this weighted sum over an interval of time
- Two contributions:
  - Construct a reduced-dimensional space in which to solve the optimization – better speed & robustness
  - Provide a physically-based keyframing tool to ensure that internal strain energies are minimized in the keyframes – better convergence & more optimal results

Reduced Simulation: How it works...

$$\ddot{q} = F(q, \dot{q}, t) + Bu$$



- q: the state vector
- F: some nonlinear function specifying dynamics of the object
- B: constant control matrix
- u: control vector

- U: orthonormal basis matrix
- q ~ Uz
- w is projected control vec

$$\ddot{z} = \tilde{F}(z, \dot{z}, t) + \tilde{B}w, \quad \text{for} \quad \tilde{F}(z, \dot{z}, t) = U^{\mathrm{T}} F(Uz, U\dot{z}, t)$$

Objective Function: What to minimize

$$E = \frac{1}{2}\sum_{i=1}^{K}\left(q(t_i) - \bar{q}_i\right)^{\mathrm{T}} Q_i \left(q(t_i) - \bar{q}_i\right) + \frac{1}{2}\sum_{i=0}^{T-1} u_i^{\mathrm{T}} R_i u_i$$

- Q: state error matrix
- R: control effort matrix
- Projection: $\bar{z}_i = U^{\mathrm{T}} M \bar{q}_i$

$$\tilde{E} = \frac{1}{2}\sum_{i=1}^{K}\left(z(t_i) - \bar{z}_i\right)^{\mathrm{T}} \tilde{Q}_i \left(z(t_i) - \bar{z}_i\right) + \frac{1}{2}\sum_{i=0}^{T-1} w_i^{\mathrm{T}} \tilde{R}_i w_i$$

Forming a low-dimensional basis

- Keyframes must have low elastic strain energy
  - Use a force (or load) based keyframing tool
  - Otherwise, if keyframes are imported, optimize them beforehand
- Construct the basis using
  - Well-formed keyframes
  - Keyframe tangents (helps with large intervals)
  - Possibly some additional deformation examples

Minimization

- Global solution cannot be guaranteed, and initial guess is important
- Use conjugate gradient descent where gradients are computed using the adjoint method
- Initial guess is provided by PD controllers for harder sequences

Results

	v	elements	T	K	r	basis time		forward pass		backward pass		total space-time		memory	
						generate	cubic poly.	red.	full	red.	full	red.	full	red.	full
dinosaur	1493	5249	120	2	25	5.5s	90s	0.036s	17s	0.066s	58s	84s	1 hour	94 KB	16 MB
flower	2713	7602	300	8	22	25s	106s	0.069s	56s	0.111s	208s	140s	24 hours	206 KB	75 MB
fish	885	3477	210	5	24	11.3s	54s	0.059s	18s	0.100s	60s	345s	3.3hours (F)	158 KB	17 MB
elephant	4736	19386	240	8	16	30s	245s	0.050s	194s	0.067s	516s	30s	17.5hours (F)	120 KB	104 MB
bridge	9244	31764	150	3	18	5.0s	mass-spring	9 s	100s	34s	130s	50min	failed (F)	84 KB	127 MB

Table 1: Optimization statistics: *v=#vertices in tetrahedral mesh, T=#frames, K=#keyframes, r=basis dimension.* F = *converged to a visibly suboptimal local minimum. Machine specs: Apple Mac Pro, 2 x 2.8 GHz Quad-Core Intel Xeon processor, 4 GB memory.*

The Graph Camera

Voicu Popescu, Paul Rosen, Nicoletta Adamo-Villani (Purdue)

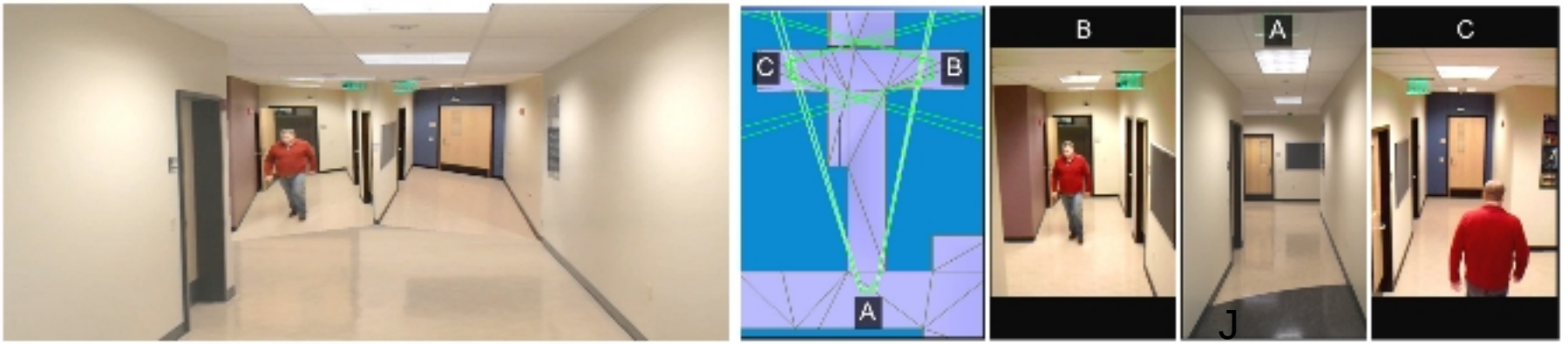


Figure 2 Single-image comprehensive visualization of real-world scenes. The graph camera image (*left*) seamlessly integrates 3 video feeds (*right*) and shows all 3 branches of the T corridor intersection.

The Graph Camera is a tool for generating multi-perspective views of a complex 3-D environment. The Graph Camera begins as a standard planar pinhole camera frustum. That frustum then undergoes a series of bending, splitting, and merging operations. The resulting images are mostly continuous and allow for comprehensive views of 3-D space.

The Graph Camera Model

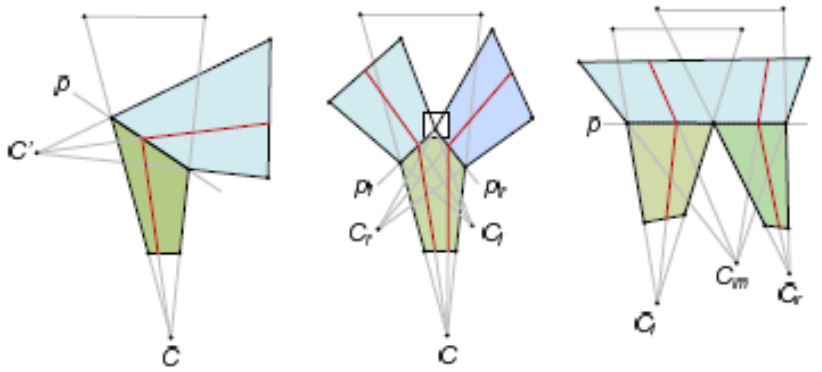


Figure 4 Basic construction operations: bending (*left*), splitting (*middle*) and merging (*right*). Sample rays are shown in red.

- Frustum Operations:
  - Bending
  - Splitting
  - Merging
- Each ray is C^0 continuous
- Renders are non-redundant

The Graph Camera Model

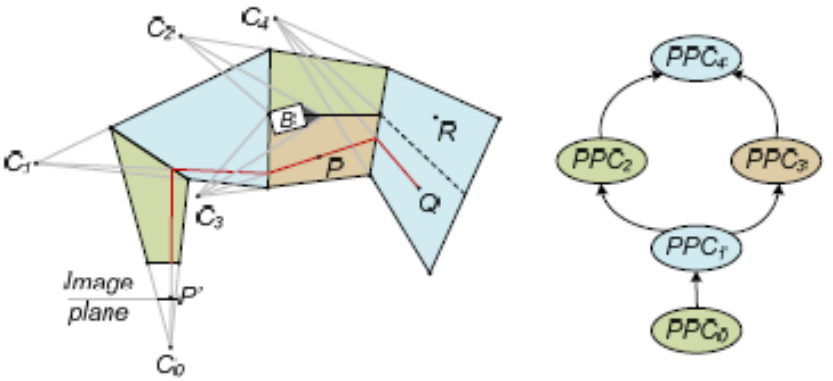


Figure 5 Graph camera with 5 PPC frusta. The PPC_1 is split into convergent PPC_2 and PPC_3 to shrink the occlusion shadow of the rectangular object B. P and Q are projected at P'.

- Can use raytracing or faster feed-forward rendering
- Given a graph camera with root frustum PPC_0 and a 3-D point P inside a frustum PPC_i
- Compute the image plane projection P' of P using a 4-D matrix:

  $$M0_i = M_0 M_1 \ldots M_i$$

  where M_k ($k = 0..i$) are the 4-D PPC matrices for the frusta on the path from PPC_0 to PPC_i
- For example point P in Figure 5 is projected with matrix $M_0 M_2 M_3$

Application Areas

- Scene Exploration
- Scene Summarization
- Real-World Scene Visualization

Scene Exploration

- Portals

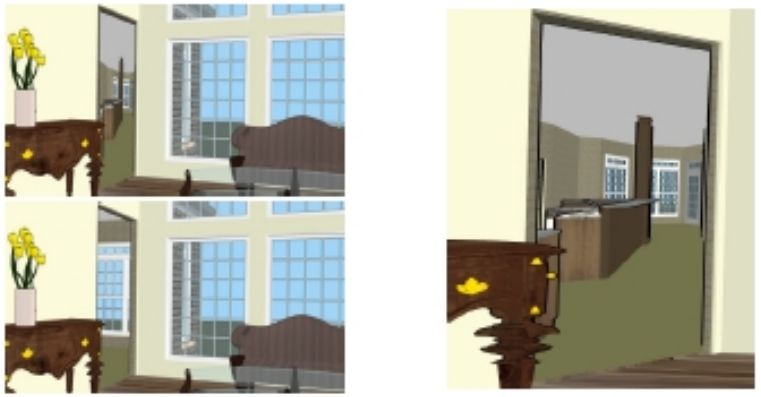


Figure 6 Portal-based graph camera image (*top left* and fragment *right*) and PPC image for comparison (*bottom left*).

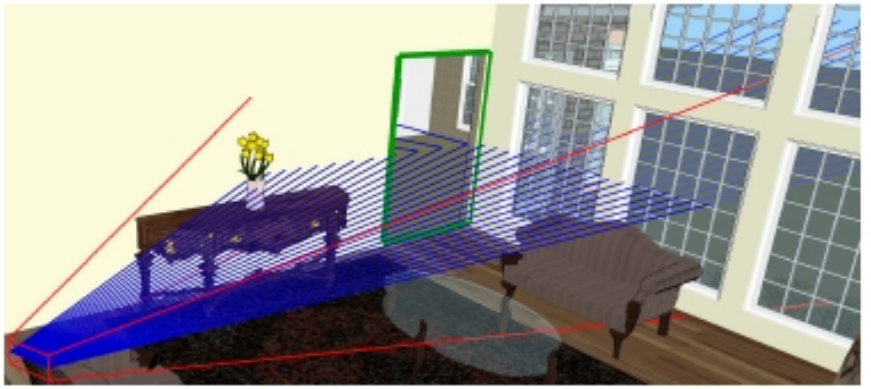


Figure 7 Visualization of portal-based graph camera from Figure 6.

- Occluders

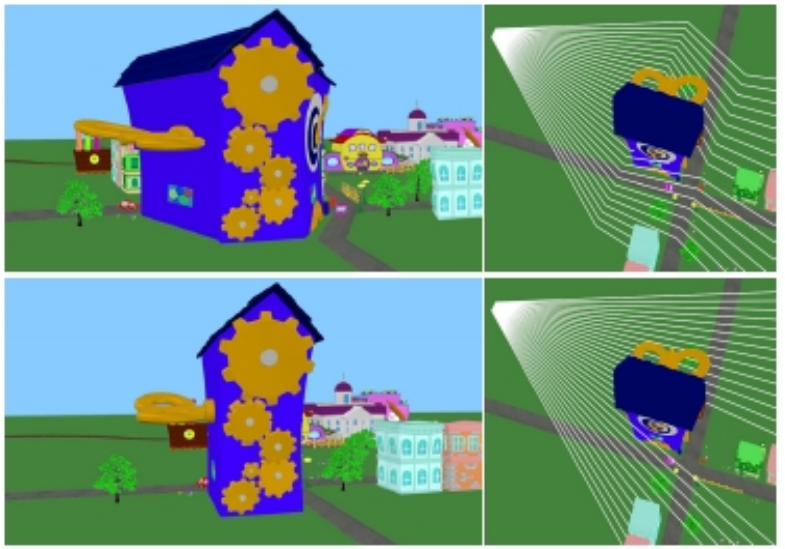


Figure 8 Occluder-based graph camera image (*top left*), PPC image for comparison (*bottom left*), and ray visualizations (*right*).

Scene Exploration

- Mazes

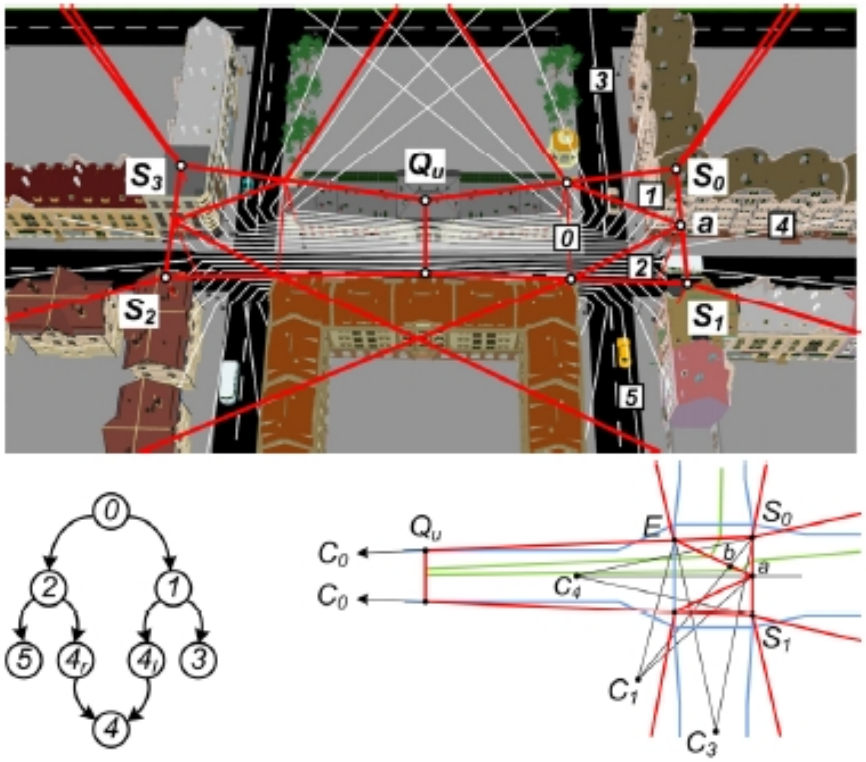


Figure 10 Visualization of forward and backward maze-based graph cameras for Figure 1 (*top*), graph of PPC frusta for forward graph camera (*bottom left*) and construction details (*bottom right*).

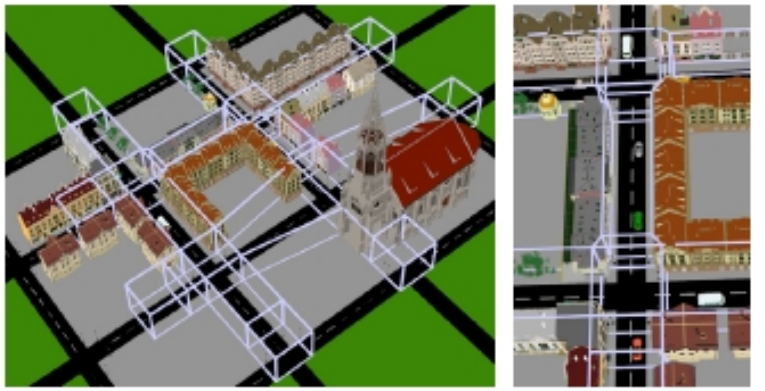


Figure 9 Visualizations of maze used to construct graph camera.

Figure 11 Top half of uneven split raw graph camera image.

Scene Summarization

Interactive Construction

Figure 12 Collage and 3-D remodeling summarization images.

Figure 13 Graph camera model visualization for Figure 3.

Figure 3 Graph camera image that summarizes a cartoon town scene (*left*) and conventional image for comparison (*right*).

Scene Summarization

Recursive Maze Construction

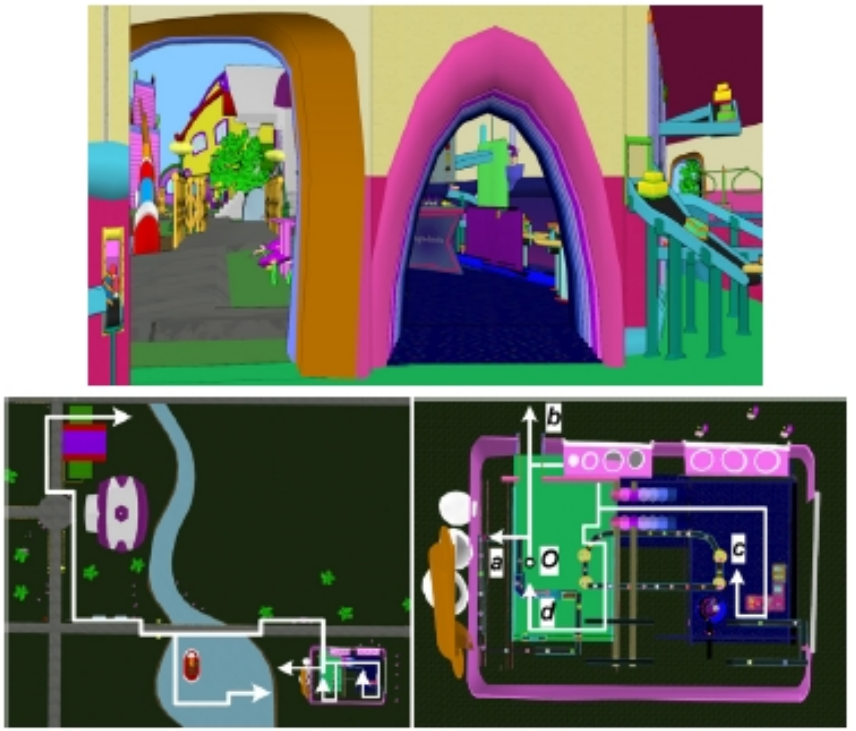


Figure 14 Summarization image rendered with a graph camera constructed with the recursive maze-based algorithm (*top*), camera model visualizations (*bottom*), overall and inside bakery.

Real-World Scene Visualization

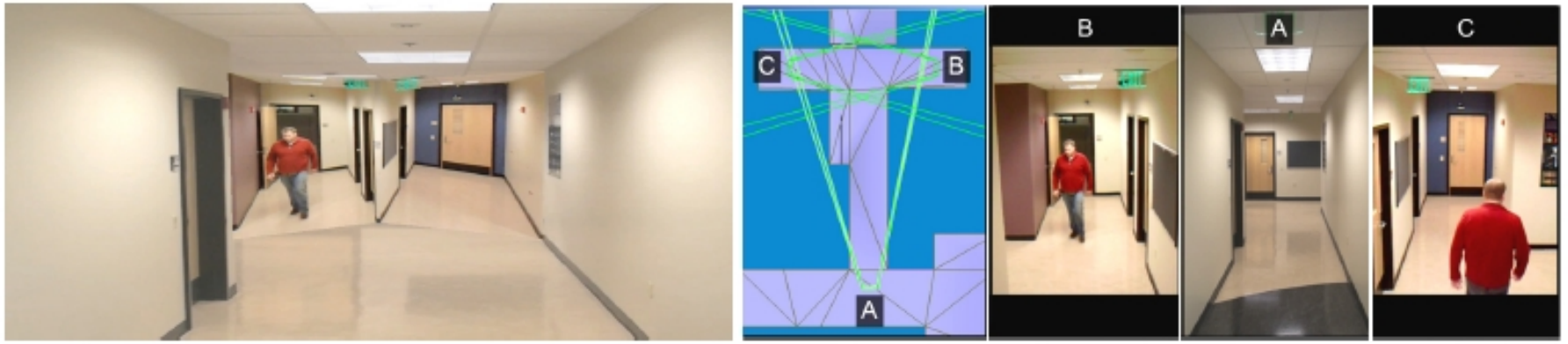


Figure 2 Single-image comprehensive visualization of real-world scenes. The graph camera image (*left*) seamlessly integrates 3 video feeds (*right*) and shows all 3 branches of the T corridor intersection.

Figure 15 Illustration of near clip plane simulation by background subtraction. The back of the subject is sampled by the video camera that is supposed to sample the right branch of the corridor (*top*). The back of the subject is erased using a background image of the right corridor acquired before the subject appeared (*bottom*).

Limitations

- No C1 continuity – abrupt perspective changes between frusta
- Maze-based constructor assumes orthogonal intersections and planar edges
- Real-world application depends on robustness of background subtraction algorithm