Algorithms and techniques for virtual camera control

Session 5: Camera path planning

M. Christie, Univ. Rennes 1
C. Lino, Univ. Rennes 1
R. Ranon, Univ. Udine

Creating camera motion

The Witcher 3 – 2015 (CD Projekt RED)

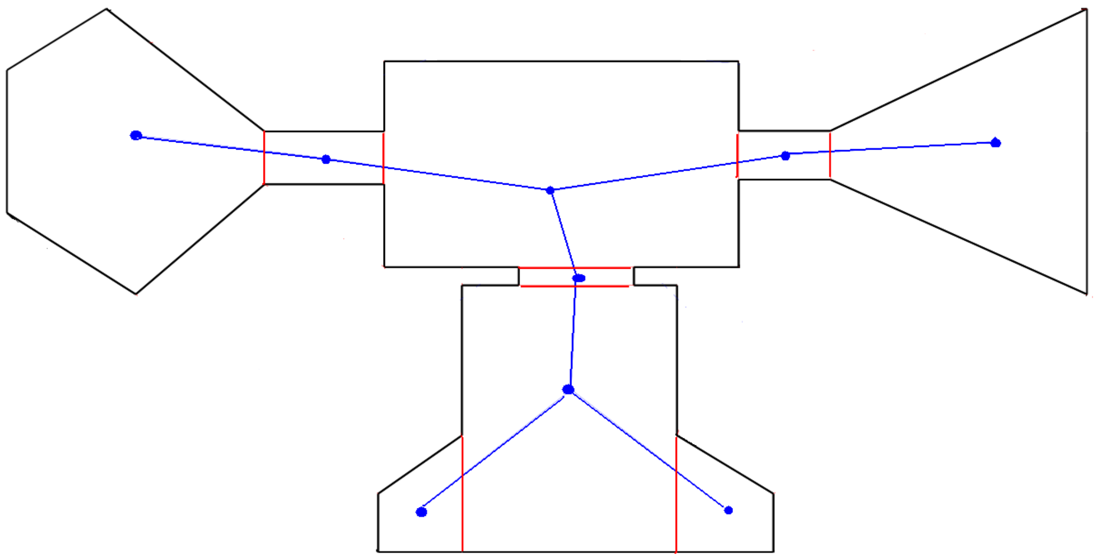
Creating (realistic) camera paths

…is a specific challenge

- it displays the issues the **current path planning techniques** have (how to decompose the environment, how to plan paths)
- and the **issues related to camera control**:
  - ensuring visual on-screen properties along the path (visibility, framing, angle, …)
  - enforcing smoothness of camera motions/orientations
  - respecting classical features of camera motions

Cell-and-portal decomposition

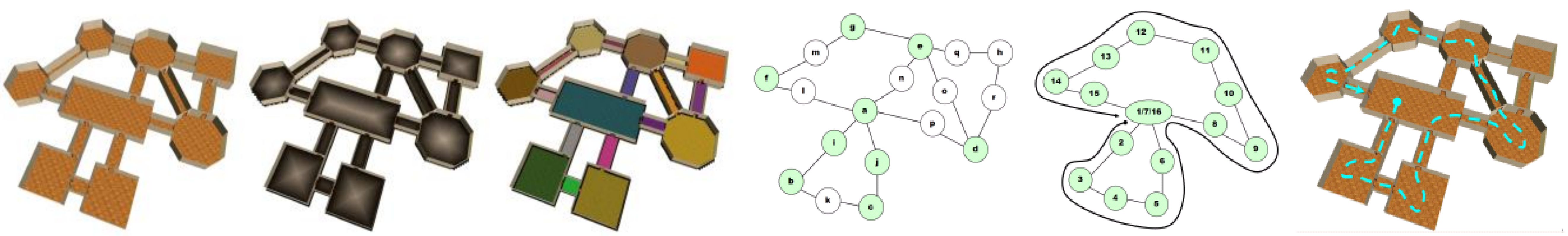
- performs partitions of the environment into sub-regions (the **cells**), and connections between sub-regions (the **portals**)



- an adjacency graph is built by connecting cells
- camera exploration/navigation tasks can then be casted as a planning process in the adjacency graph [AVF04]

Cell-and-portal decomposition

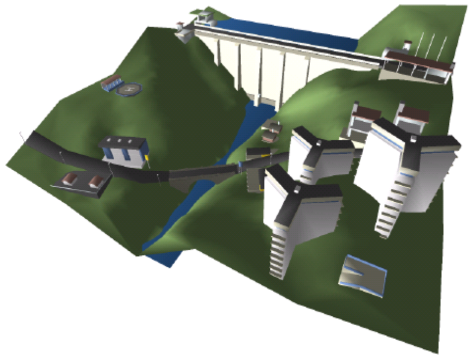
- provides a *structure* to the environment to better perform navigation/walkthrough tasks (the decomposition can be authored)
- Andujar etal. [AVF04] employ this structure to:
  - identify the individual interest of each cell (with an entropy-based metric)
  - compute the sequence of most relevant cells to visit
  - compute a path connecting the cells, portals and relevant viewpoints in the cells

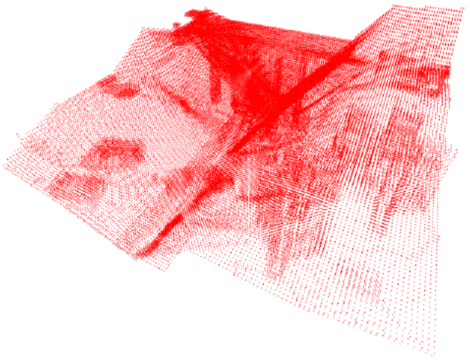


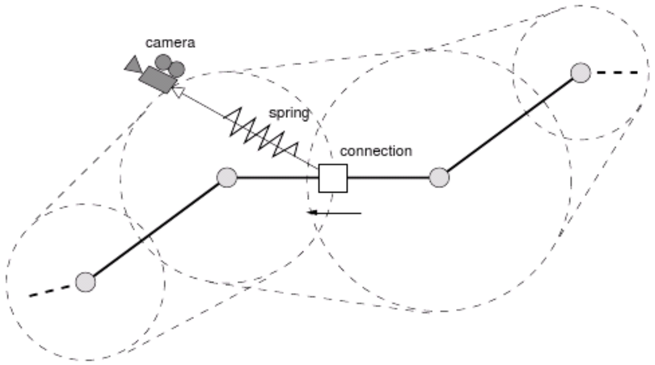
(a) (b) (c) (d) (e) (f)

Voxel-based decomposition

- a regular partitioning of the free space (voxels) can be used to generate guided tours [ETT07]:
  - visibility of (authored) landmarks is computed for each of the voxels in a pre-process
  - all voxels that view at least one landmark are connected together to form a adjacency graph
  - a solving process (Travel Salesman-like) computes the suite of voxels to visit in the graph to ensure that each landmark has been viewed at least once
  - in interactive mode, a memory of the visited landmarks is maintained to guide/constrain the users navigation, through a spring-based physical system

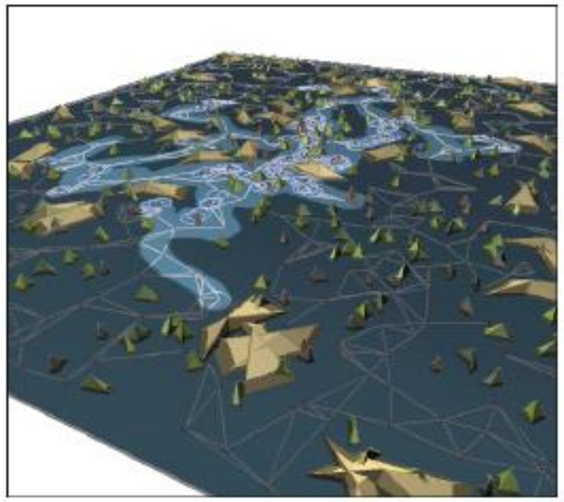


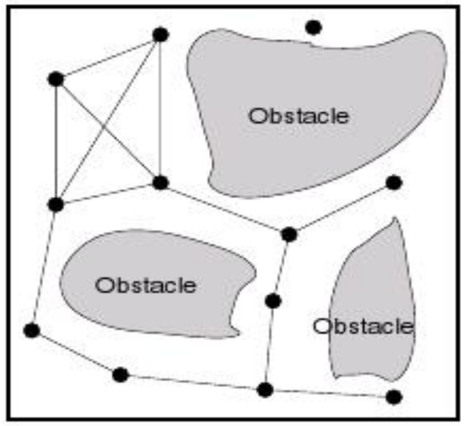




Roadmap constraints

- roadmap planners operate in two phases:
  - first sampling the space of possible configurations
  - second constructing a connectivity graph by linking neighbour samples (and checking for collision on the links)

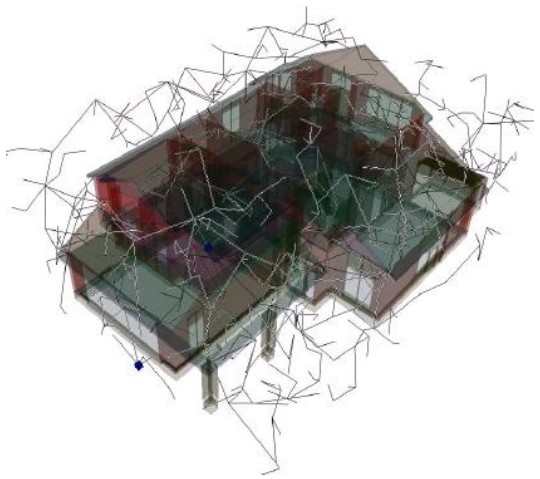


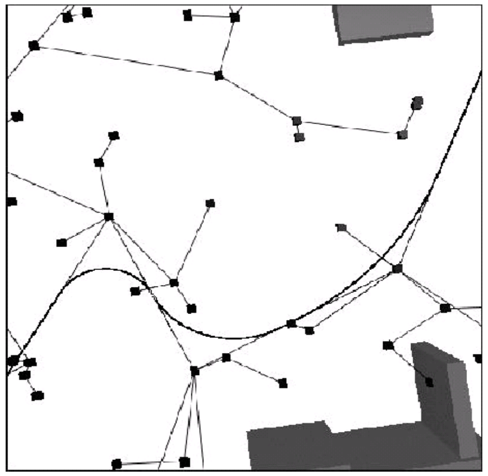


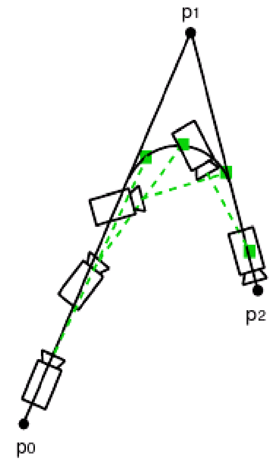
  - simple to construct and navigate inside the graph
  - complex to determine the appropriate density of sampling (but PRM complexity is a factor of the scene complexity)

Roadmaps in camera planning

- [NO03] rely on probabilistic roadmap techniques for camera planning:
  - roadmap is consisting of collision-free camera motions (the camera is abstracted as a sphere, the motion as a cylinder)
  - planning is performed with an advanced Dijkstra process (avoids sharps turns)
  - path is smoothed and camera orientation anticipates camera motion

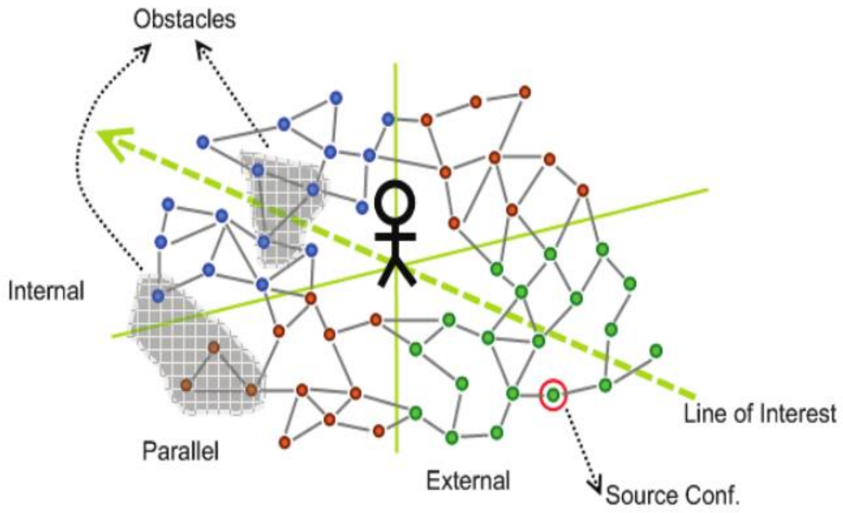






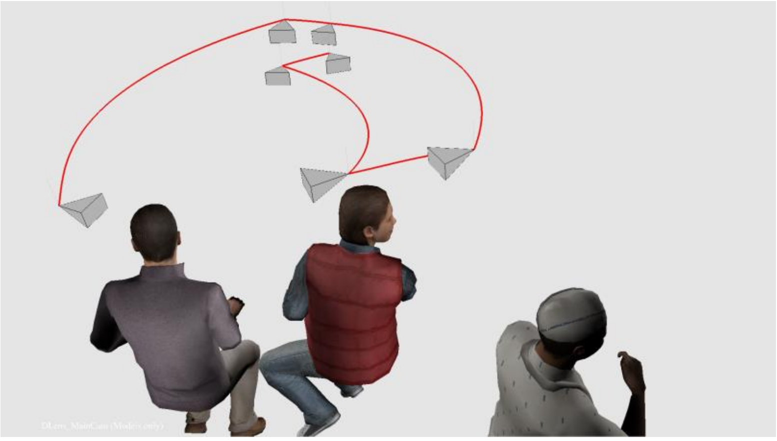
A local/dynamic roadmap

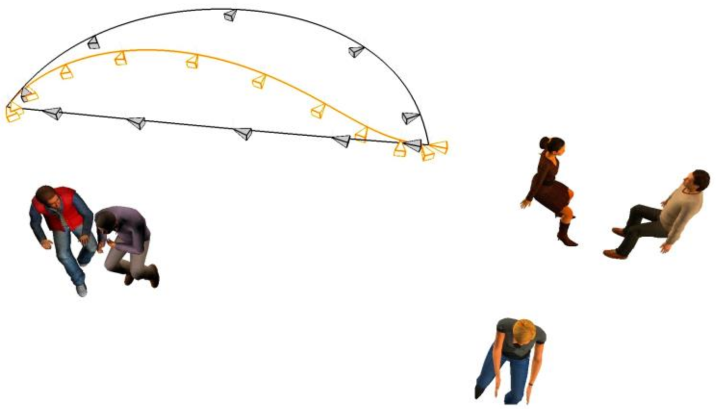
- Using a locally defined probabilistic roadmap [LC08]
  - a probabilistic roadmap is created around the target and moves with the target (camera positions are expressed in polar coordinates)
  - the path planning is performed in the roadmap to move the camera
    - collision/occluded nodes are removed from the graph using a lazy evaluations strategy
    - new nodes are inserted using a density parameter
  - cuts can be performed between regions (by connecting distant edges)



Toric Space interpolations

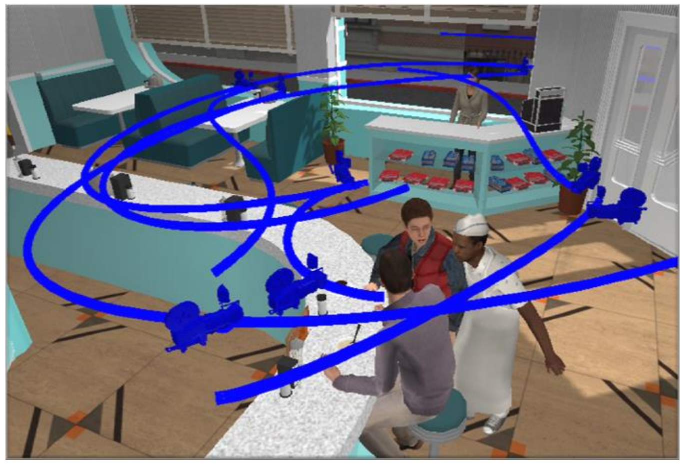
- Interpolating in the space of visual features
  - introduced by [LC15]
- given two viewpoints v1 and v2:
  - extract visual features (angle between targets, distance to targets, vantage angle of targets) for viewpoint v1 and v2
  - perform a linear interpolation of the visual features of the first framing between v1 and v2
  - perform a linear interpolation of the visual features of the last framing between v1 and v2
  - and then blend between the two trajectories

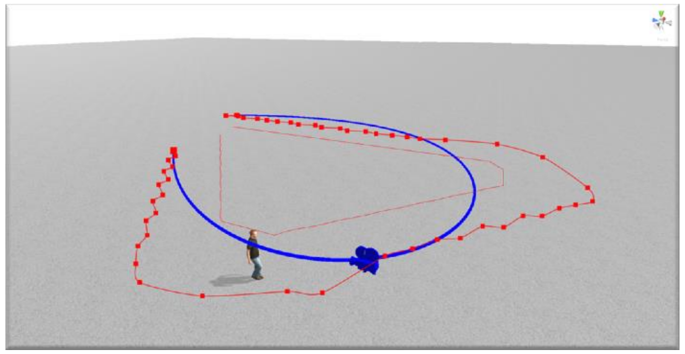


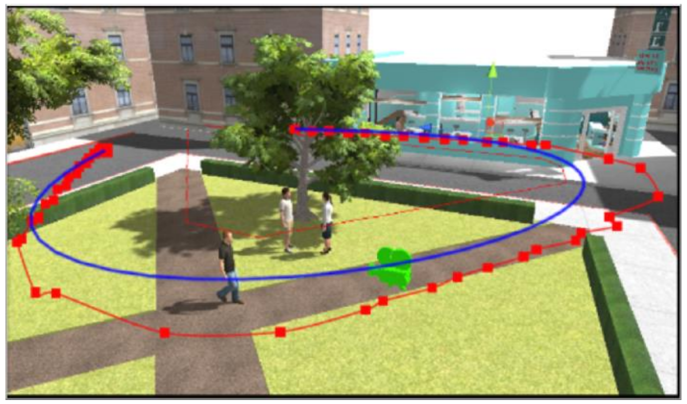


Camera-on-rails

- Back to the roots of cinematography
- given two camera configurations:
  - Extract and smooth targets trajectory
  - Compute a raw trajectory by linearly interpolating parameters of the manifold space
  - Approximate the trajectory with a virtual rail using bezier curve fitting
  - Compute the optimal positions on the rail
  - Optimize the position and orientation on the rail regarding velocity and acceleration constraints



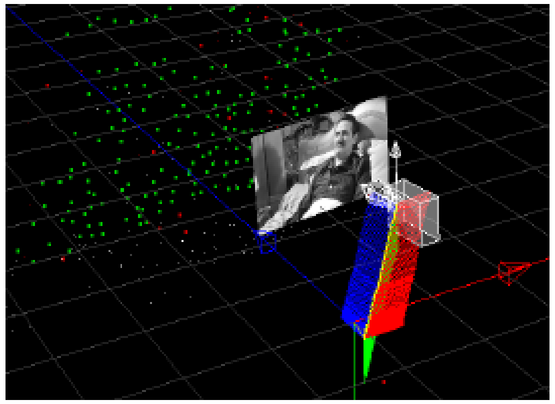




Camera-on-rails

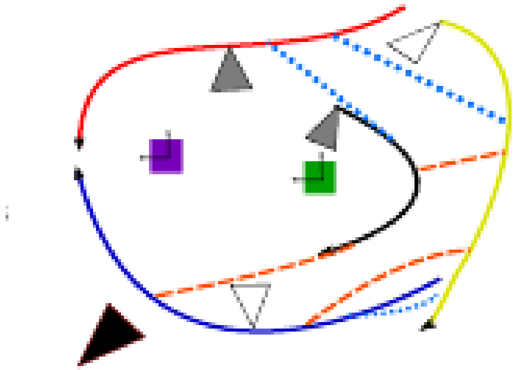
Extract and re-target camera motions

- [SDM14] propose to extract camera targets from movies
  - eg using SIFT-based feature tracking (Voodoo software)



- Trajectories are then retargeted to the virtual environment (using the ToricSpace)

- All trajectories are then expressed in a motion graph around the targets (similar to [LC08])
  - the graph enables continuous or cut transitions between pieces of trajectories
  - characteristic noise and nature of motions in maintained



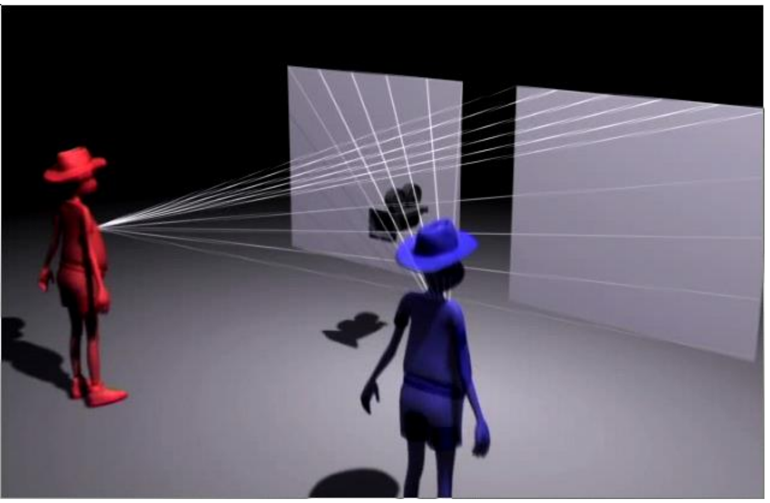
Visibility: A Fundamental Challenge

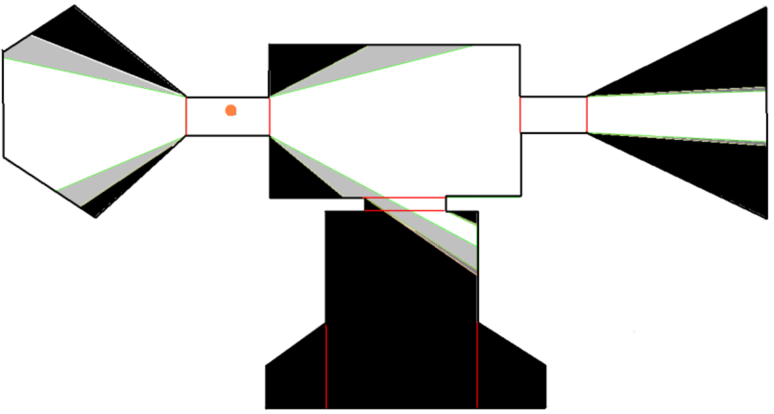
- many applications require the visibility of target objects (games, sci. visualization,...)
- importance of visibility (triggers interaction, depth cue, scene understanding, spatial relations...)
- visibility is **application-dependent**
  - a matter of perception (e.g. object recognition)
- visibility has multiple **interpretations**
  - spatial visibility (considering sparse occluders)
  - temporal visibility (with fast moving occluders)

And Complex Challenge

- two problems:
  - how visible is the target?
  - where should I move the camera to?
- cost of evaluating visibility/predicting motion
  - complexity of the target/complexity of the scene
  - maintenance of visibility data structures
- maintaining visual stability with **sparse or fast-moving** occluders
- integration of visibility computation in the whole camera control process
  - how to balance its influence with other descriptors

Handling Visibility



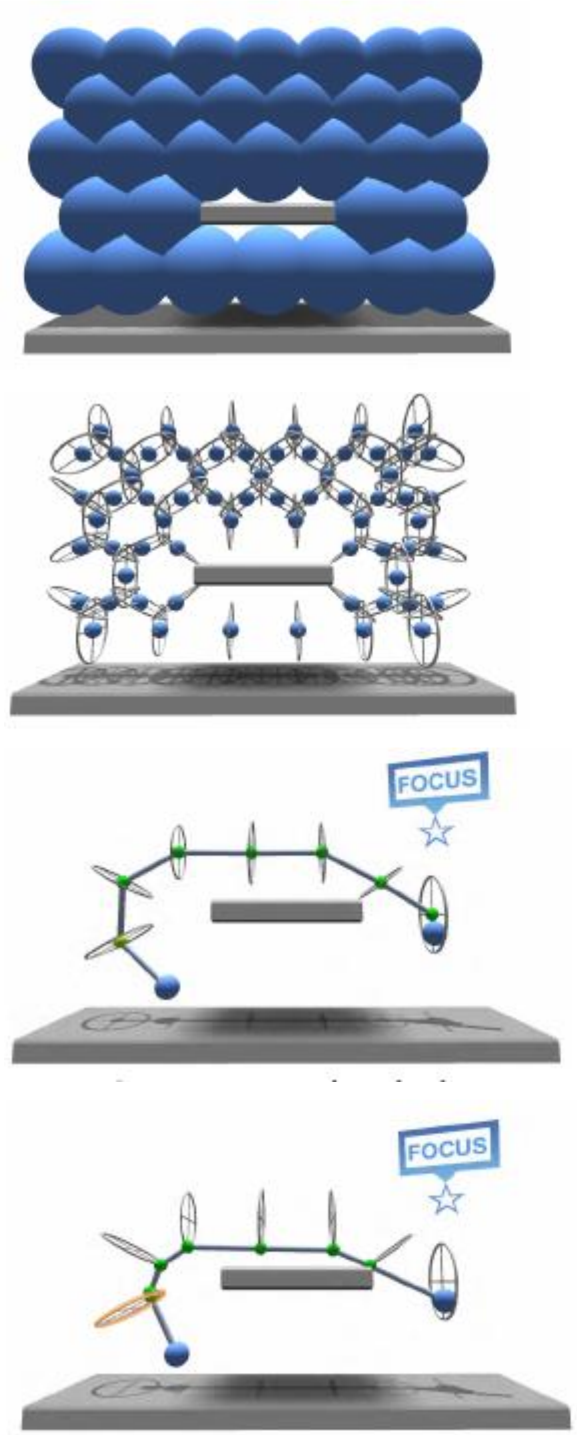


Two classes of techniques for camera control:

- local visibility computation:
  - principle: **sample** or **reason** in a local area
    - with ray-casting techniques
    - with bounding volume intersection
    - with hardware rendering techniques
- global visibility computation:
  - pre-computation from the static geometry (offline)
    - cell-and-portal visibility structure
    - hierarchical cells, ...
  - Followed by an online estimation of visibility

A global/dynamic roadmap

- Oskam et al [OSTG10] propose a visibility aware roadmap technique:
  - uniform sphere-sampling of the free space in the environment
  - pre-computing sphere-to-sphere visibility (stochastic ray casting)
  - connecting overlapping spheres to build a roadmap
  - planning a rough path from source to target that ensure visibility of a target (focus point)
  - refining the path using rendering-based technique



A global/dynamic roadmap

Discussion over local visibility techniques

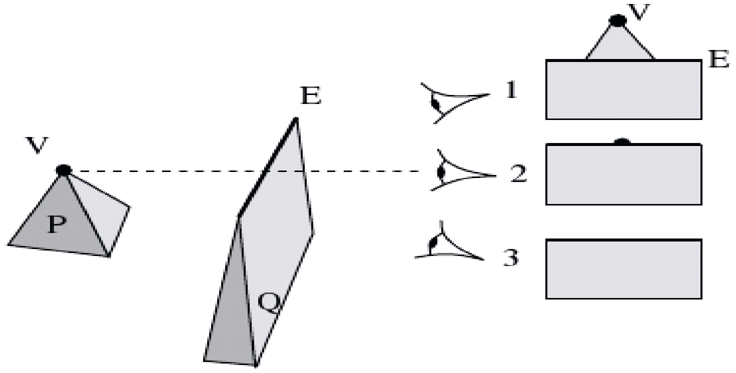
- **simple** to implement and **efficient**
- CPU/GPU-adaptive (ray-casting or frame rendering)
- adapted to **dynamic** environments

But: lacks global visibility

- leads to issues in local minima areas
- inappropriate for performing cuts between shots

Global visibility techniques

- provides a collection of techniques and structures to represent the visibility in an environment:
  - grounded on the notion of *visual events*



  - *a visual event* separates the space into visible and non-visible areas

  - two classes of problems are considered in the literature
    - *from-point* visibility computation
    - *from-region* visibility computation

Discussion

Handling visibility remains a complex topic:

- cost for precise/complete evaluation of visibility of complex/multiple targets
- strong link with planning techniques
- necessity of coupling of local and global visibility techniques
- importance of anticipating actions/motions
- importance of studying the nature of the targets and occluders

Bibliography

- [AVF04] Aandujar C. G., Vazquez P. P. A., Fairen M. G.: *Way-finder: Guided tours through complex walkthrough models*. Proceedings of the Eurographics Conference, 2004.
- [Bec02] Beckhaus S.: *Dynamic Potential Fields for Guided Exlporation in Virtual Environments*. PhD thesis, University of Magdeburg, 2002.
- [HMK∗ 97] Hong L., Muraki S., Kaufman A., Bartz D., He T.: *Virtual voyage: interactive navigation in the human colon*. In Proceedings of the Siggraph Conference, 1997.
- [NO03] Nieuwenhuisen D., Overmars M. H.: *Motion Planning for Camera Movements in Virtual Environments.* Tech. Rep, Utrecht University, 2003.
- [HHS01] Halper N., Helbing R., Strothotte T.: *A camera engine for computer games: Managing the trade-off between constraint satisfaction and frame coherence*. In Proc. of the Eurographics Conference, 2001.
- [LCF10] Lino, Christie, Lamarche, Schofield, Olivier: *A real-time cinematography system for interactive 3d environments*; Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer (SCA'2010)
- [OSTG09] Oskam, Sumner, Thuerey, Gross. 2009. "*Visibility transition planning for dynamic camera control*". In Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '09)
- [CDM14] Sanokho CB, Desoche C, Merabti B, Li TY , Christie M, "*Camera Motion Graphs*", Proceedings of the 2014 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '14)
- [LC15] Lino C. and Christie, M. "*Intuitive and Efficient Camera Control with the Toric Space*" Proceedings of SIGGRAPH 2015.
- [CNO12] Christie M, Normand JM, Olivier P, *"Occlusion-free camera control for multiple targets",* Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '12)