Unity for movie production

Marc Christie - Marc.Christie@irisa.fr
Univ Rennes, INRIA - France

Unity content from Mathieu Muller
Mathieu@unity3d.com (Product Manager Film & TV)

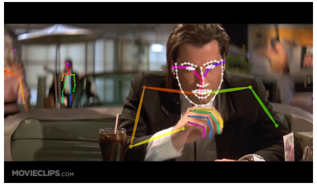
unity

Who am I?

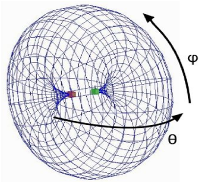
- Associate Professor in CS, University of Rennes 1, INRIA, France

Research interests:

- Real and Virtual cinematography



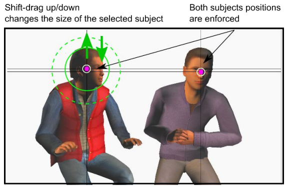
Understand



Formalize

$$\forall t, \exists \mathbf{x} | f(t, \mathbf{x}) = 0$$

Solve



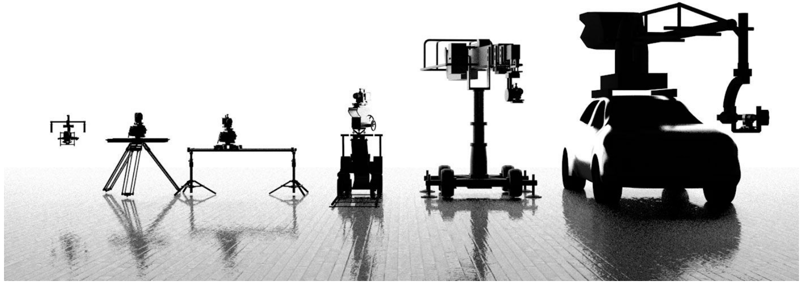
Interact

- *Easing the control* over virtual cinematography
- Applications to games, interactive narratives, movie pre-production

What Is Cinematography?

- Both a **Techniques and an Art** for storytelling, related to
  - How a shot is composed in the screen space
  - How a camera moves along time
  - How shots are edited together to create a sequence

- Cinematography also encompasses
  - How lights are positioned
  - How the decor is spatially arranged
  - How the staging is orchestrated (mise-en-scène)



- **Virtual Cinematography**
  - Transposition / Adaptation of existing techniques to virtual 3D environments

Virtual Cinematography / Games / Storytelling

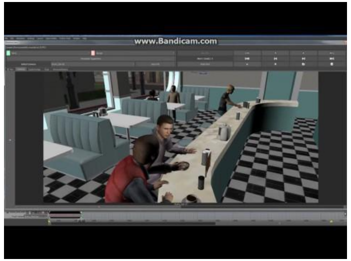
Heavy Rain © Quantic Dream, 2010

Virtual Cinematography / GAMES / Storytelling

The Witcher 3 © CD Projekt, 2016

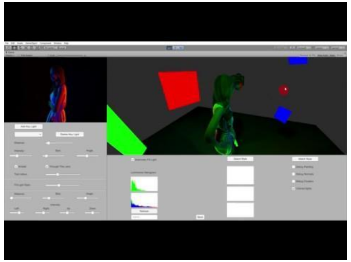
My Research Topics?

Interactive and automated cinematography

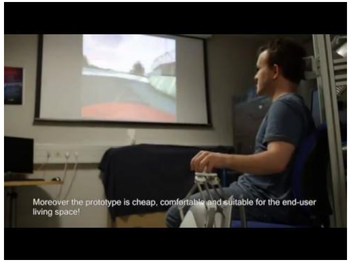


Analyzing film patterns

Interactive Lighting



Haptic cinematography



Film Production Pipeline

Preproduction		Production		Postproduction

Previs	Techvis	Onset previs	Postvis




Previsualisation

- Rehearsing a movie in CG before shooting it

From Halon previsualization (USA)

Using Game Engines in Film Production Pipeline

Preproduction		Production	Postproduction
Previs	Techvis	Onset previs	Postvis
Modelling	Potential use of Game Engines in Production		




Previsualisation

Using Unity3D As a Film Production Tool

Photogrammetry

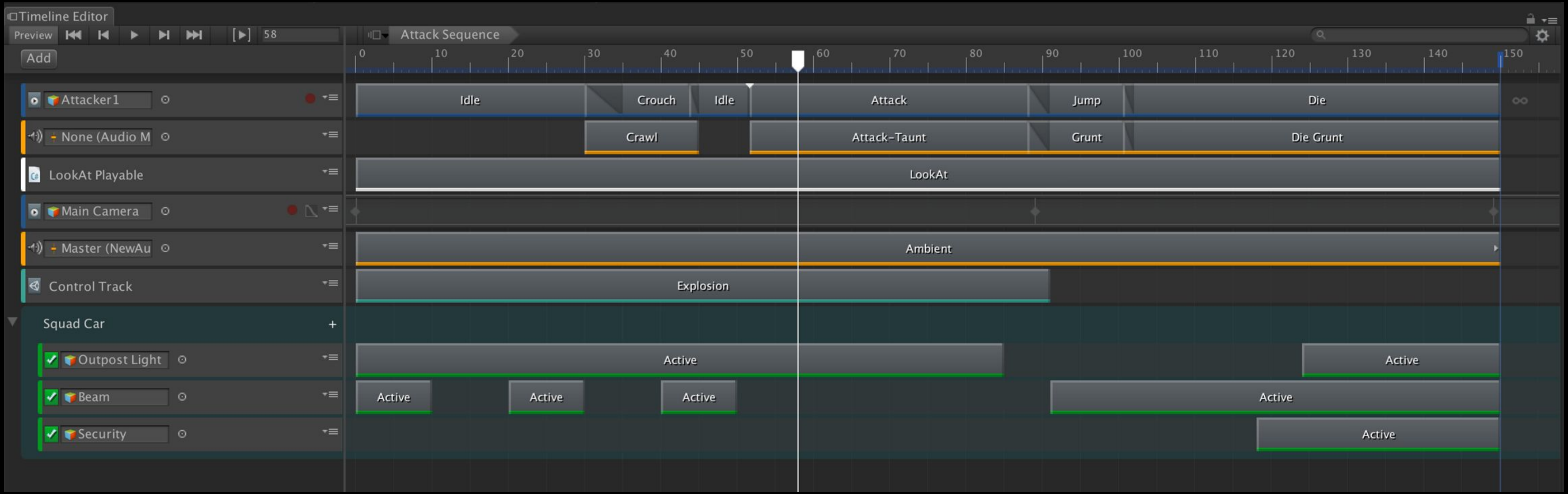
https://blogs.unity3d.com/2018/03/12/photogrammetry-in-unity-making-real-world-objects-into-digital-assets/

Photogrammetry

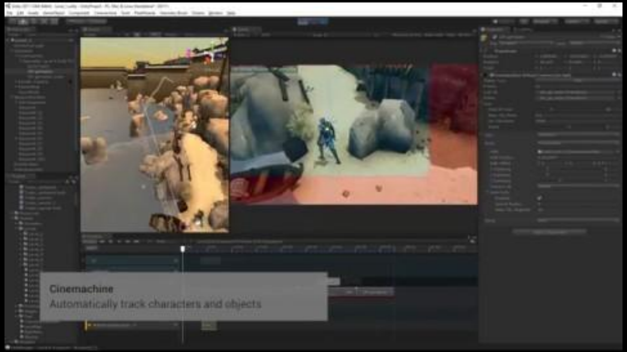
https://blogs.unity3d.com/2018/03/12/photogrammetry-in-unity-making-real-world-objects-into-digital-assets/

unity

Sequencing


Timeline Editor
Preview
58
Attack Sequence
Add
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
Attacker1
None (Audio M
LookAt Playable
Main Camera
Master (NewAu
Control Track
Squad Car
Outpost Light
Beam
Security
Idle
Crouch
Idle
Attack
Jump
Die
Crawl
Attack-Taunt
Grunt
Die Grunt
LookAt
Ambient
Explosion
Active
Active
Active
Active
Active
Active
Active

Cinemachine



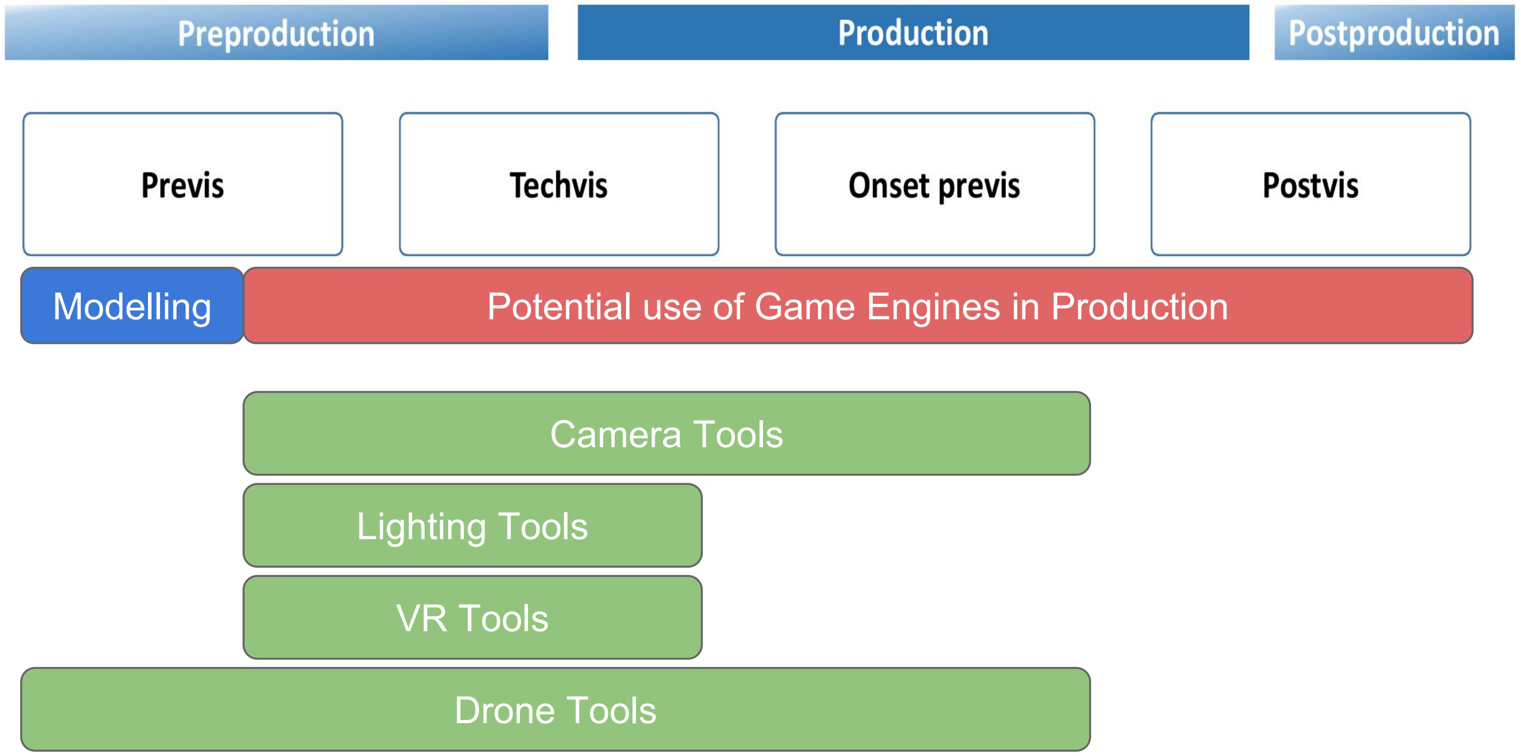
https://unity3d.com/learn/tutorials/topics/animation/using-cinemachine-getting-started
(Available as package)

unity

Performance capture

https://blogs.unity3d.com/2018/01/16/arkit-remote-now-with-face-tracking/
https://blogs.unity3d.com/2018/08/13/facial-ar-remote-animating-with-ar/

Our Tools for Virtual Production



Part 1 - Our Camera Tools

Limited controls on camera tools

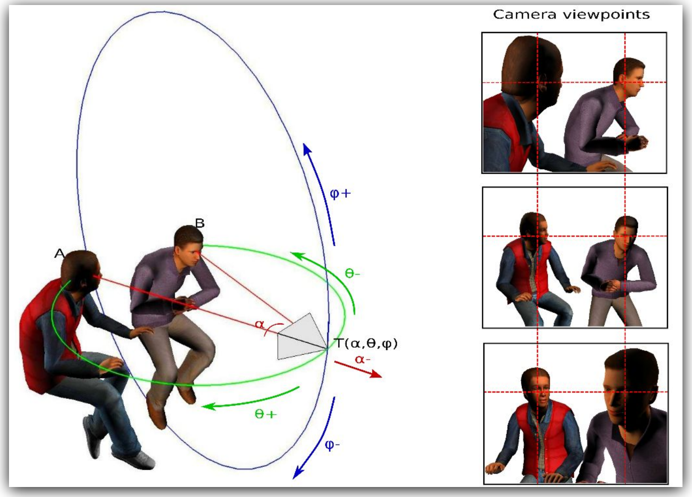
- Existing modelling tools offer limited cinematographic control
  - **Direct control of camera parameters (pos/orient/zoom/focus)**
  - **No motion pimitives (dolly/pan/zoom)**
  - **No camera rigs (crane/boom/…)**
  - **No automation (target following)**

A cinematograhic approach: Manifold Surface

- Introducing a novel 3DOF representation of a camera [LC15]
  - dedicated to viewpoint manipulation of two targets
- Three parameters to control the position:
  - α : angle between targets A and B
  - θ : horizontal angle
  - φ : vertical angle
  - the framing of the two targets is implicitly defined in the model

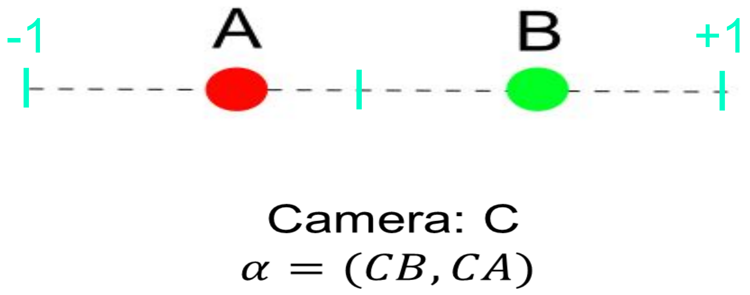
(Unity and C++ code available: ToricCam)

https://sourceforge.net/projects/toric-cam/

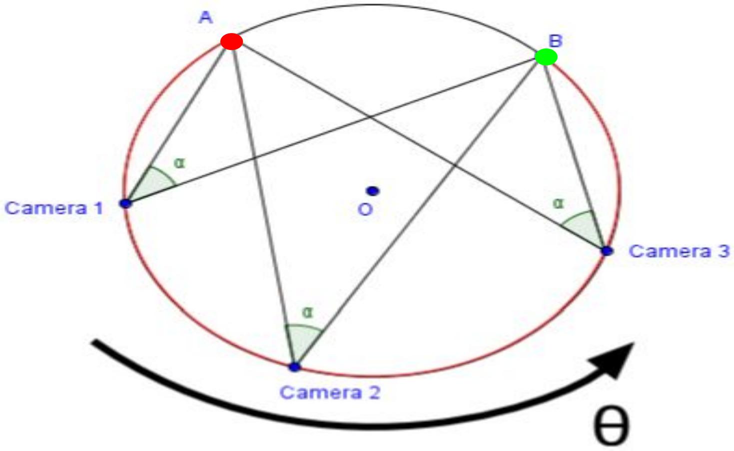


Composition: 2D intuition

Desired on-screen
Composition
(1D)



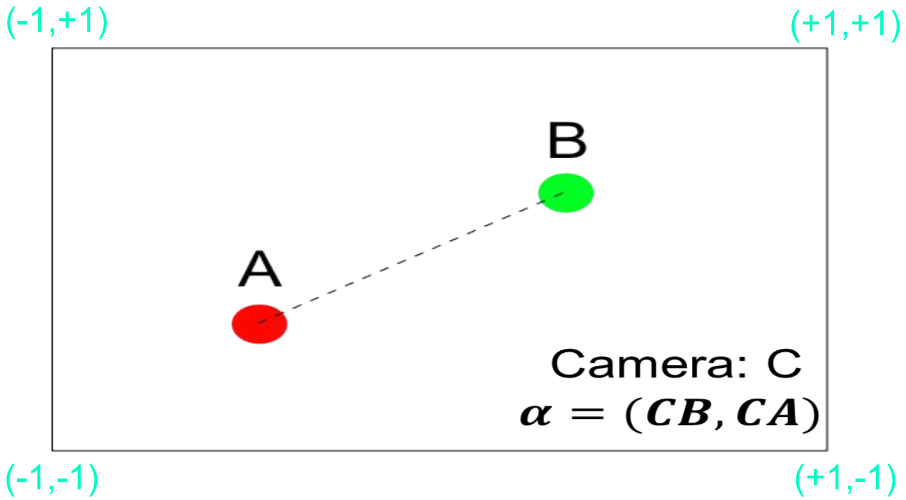
Solution = 1D parametric
manifold (θ)



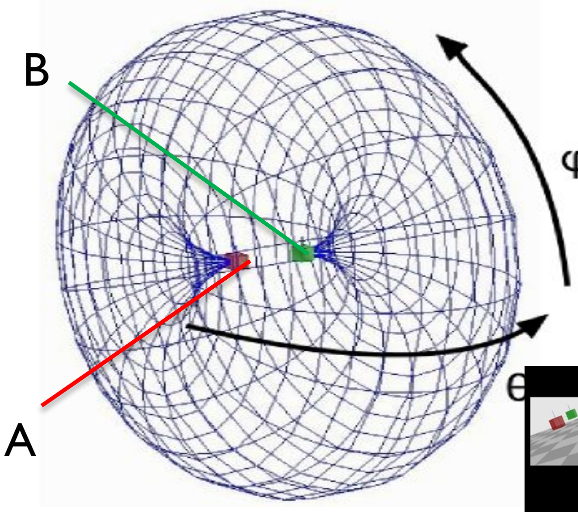
Any configuration c(θ) satisfies the 1D composition

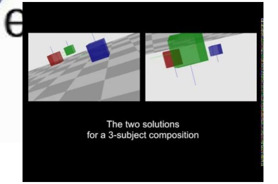
Composition: 3D environment

Desired on-screen
Composition
(2D)



Solution = 2D manifold
surface (θ,φ)
(subset of a spindle torus)





Any configuration c(θ,ϕ) satisfies the 2D composition

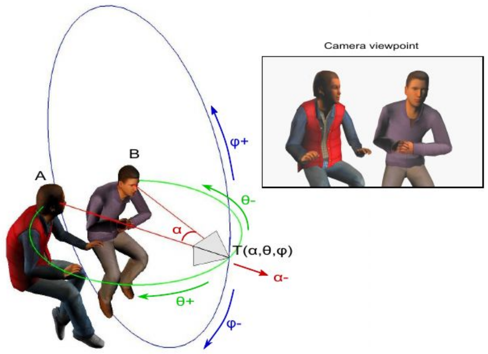
Extension: 3D Toric space

- More evolved problems:

⇒ **relax the positioning constraint**

- Generalized model of camera:
    - **3-parametric space** (α, θ, φ)

- Defines the **range of all possible manifolds** around two targets



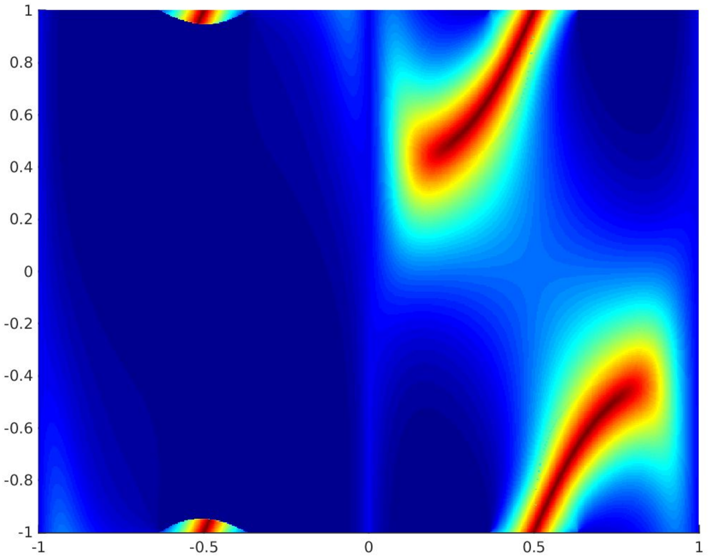
(Algebraically) casts 7D camera problems to 3D

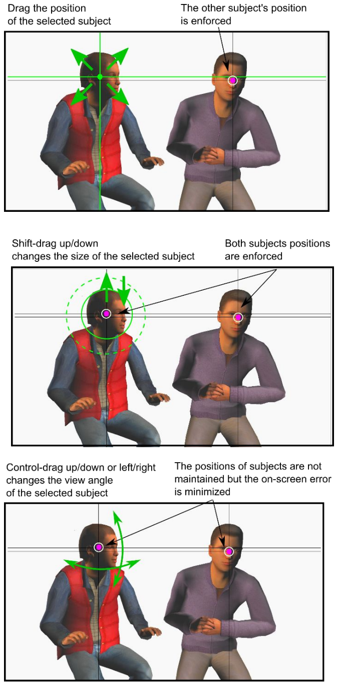
Manipulations in the Toric Space

Manipulations in the Toric Space

Principle:

- *Manipulati*
  - *while the* *-space*
  - *and roll is* *)*
- *Interaction*
  - *change o* *vantage angles*
  - *example*
    - *we search for a position on the manifold surface where roll is null and minimizes the change in on-screen position*



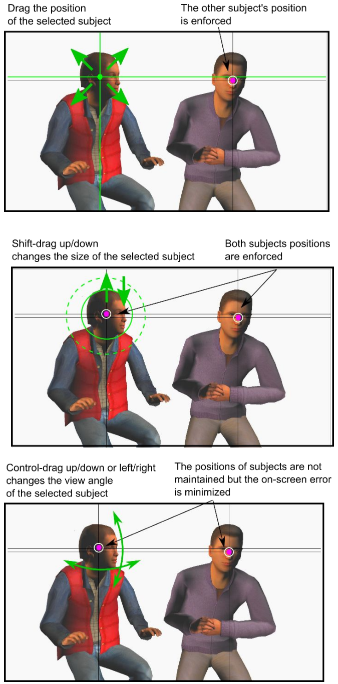


$$\min_{(\theta,\varphi)} \left(p_A - p'_A\right)^2 + \left(p_B - p'_B\right)^2$$

Manipulations in the Toric Space

Can we generalize to more targets?

- *3 targets is the well known P3P problem, for which there are only 4 solutions (most with a roll different than 0)*
- *4 and more is an over-constrained problem*



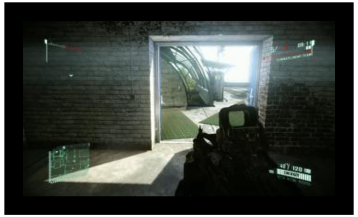
Part 2 - Our Lighting Tools

Motivations

- Lighting in 3D scenes is essential
  - Photography, Cinematography, Games
  - Sets the mood of the scene
  - Helps conveying emotions

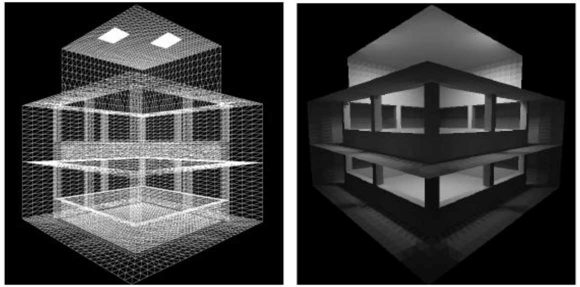
- Towards real-time realistic lighting
  - Lots of research dedicated to accurately reproduce lighting
  - Realtime area lights

- Raised the need for smarter and faster lighting tools
  - Rely on common lighting methods



- Standard control of the lighting remains tedious
  - Each light and each parameter are controlled separately
  - Requires advanced skills and expertise
- Advanced techniques are computationally expensive or limited to point-lights
  - Mostly not realtime
  - Not based on existing studio lighting models

- The famous “inverse lighting design” problem
  - Given a desired lighting (image, painting, or overlays)
  - Compute light parameters to match desired lighting
- Using optimization+rendering cycles [Kawai93]
  - render > measure > change parameters
  - optimize light positions and directions
  - using a hierarchical radiosity solver



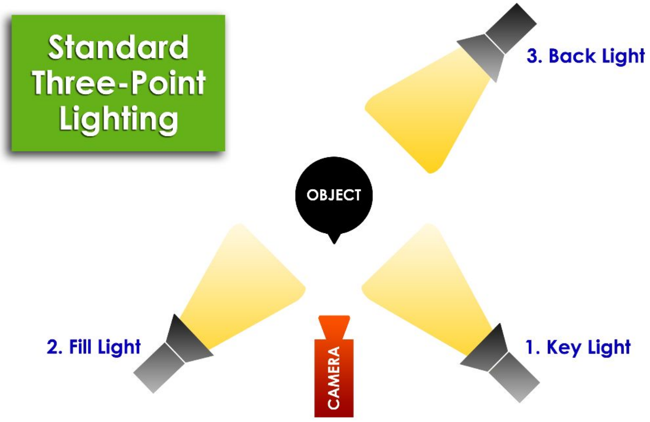
(a) Two emitters in the ceiling. (b) Reflected radiosity in patio.

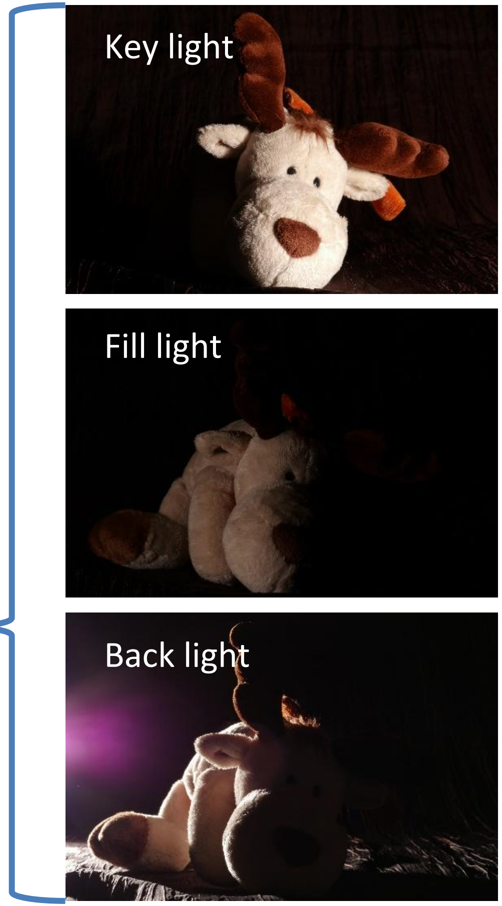
- The approach has inspired many, but remains expensive

➢3 Point lighting : a well-known lighting principle

- Key light : main light source
- Fill light : removes remaining shadows
- Back light : detour the subject

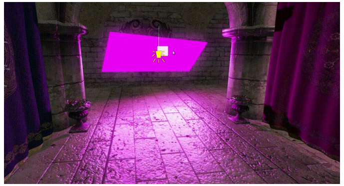
=>There are implicit relations between these lights





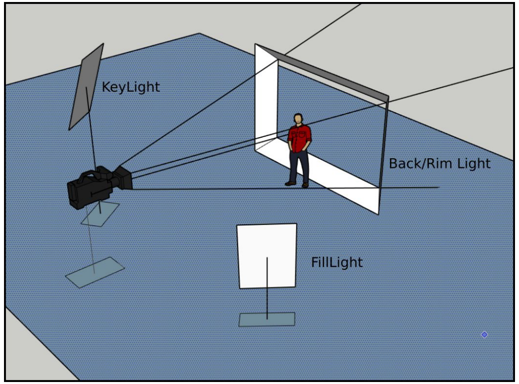
➢Light sources

- Real-time rectangular area lights oriented towards the subject



Stages:

1 - The user controls the **key-light**



2 - We automatically place **fill-lights** wrt. target

- Potentially multiple lights designed to brighten dark areas

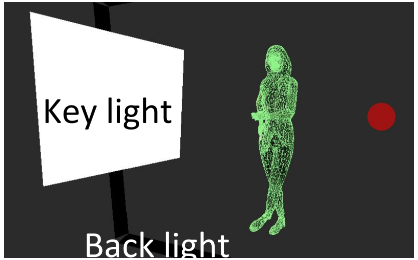
2 - We automatically place the **back-lights**

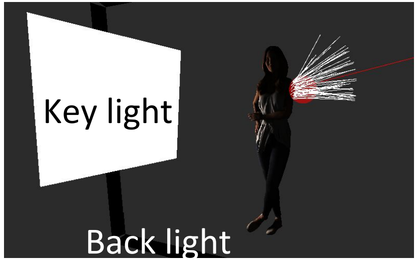
- Stays attached to the camera frustum
- Rig placed behind the actor

Light type	Manipulation steps
key-light	$d, \theta, \varphi, a, s, \lambda, f, c$
fill-light	λ, c
rim-light	c
all rim-lights	d, σ, λ

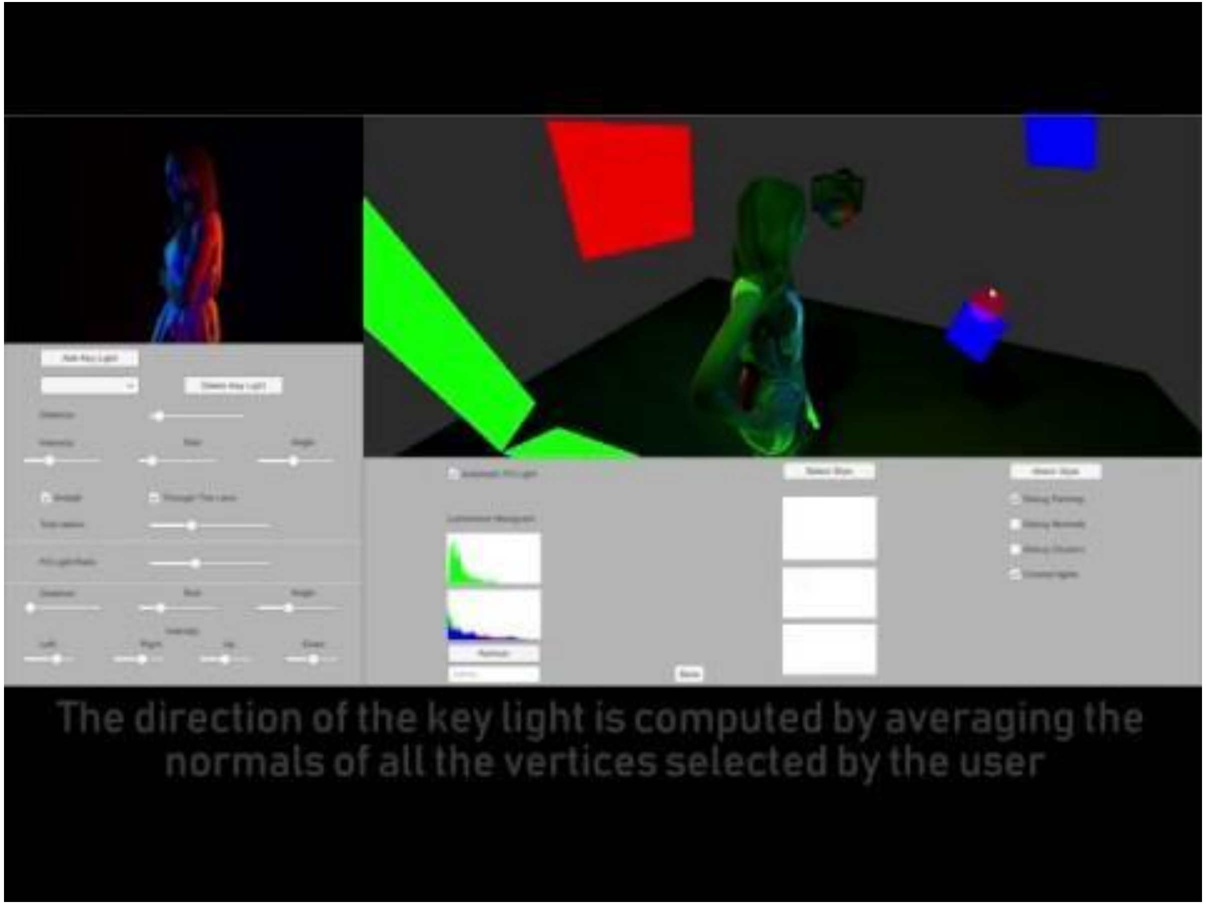
Controlling the key-light

- ➢An extended arc-ball controller on the surface
  - Process the geometry of the mesh
  - Extract the normals of the highlighted vertices
  - Average and compute the position of the desired key-light



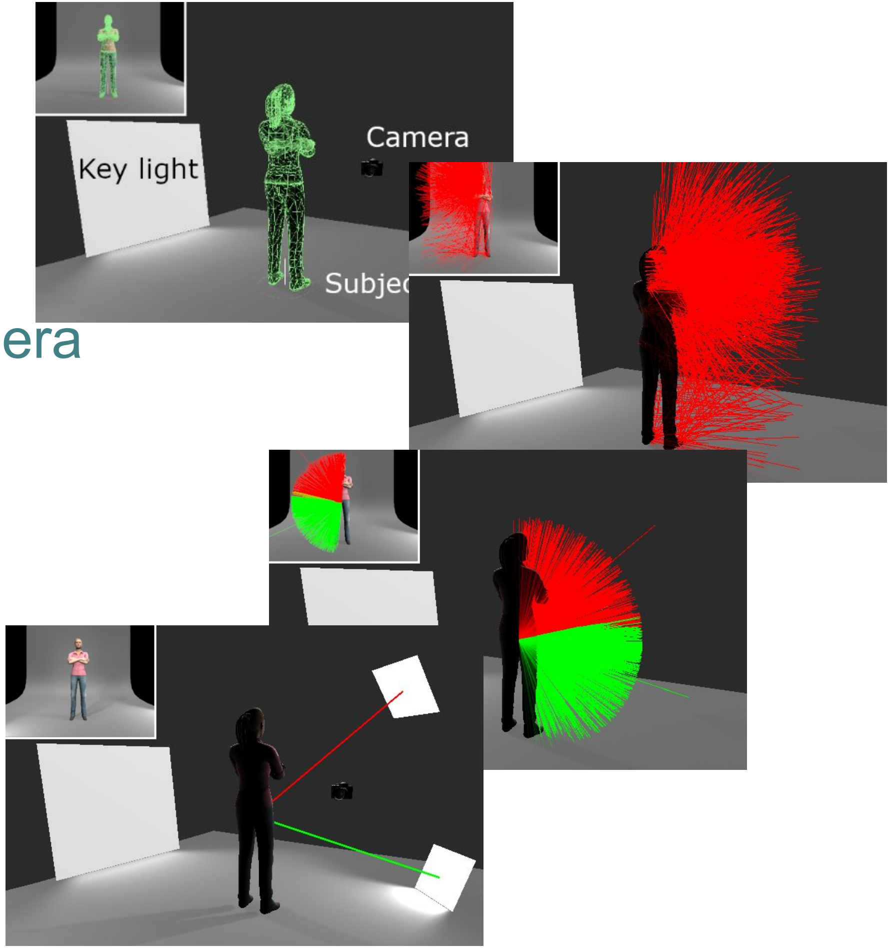


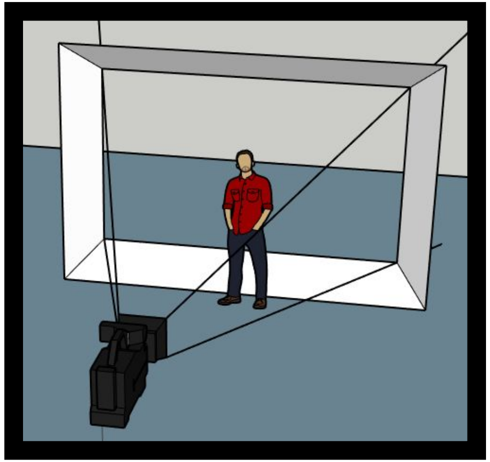
Video


The direction of the key light is computed by averaging the normals of all the vertices selected by the user

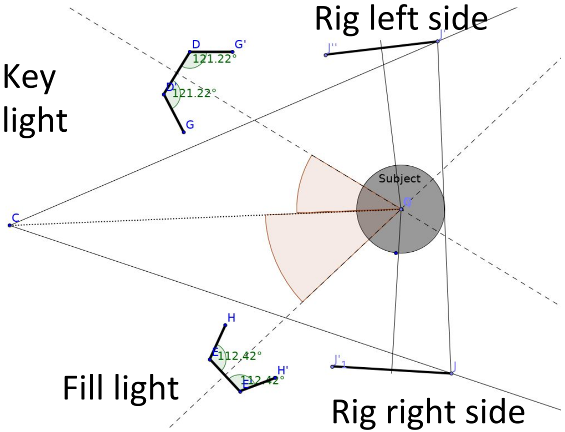
Placing fill lights

- Mesh processing
- Normals extraction from
  - Unlit vertices and
  - Vertices visible from the camera
- Clustering of the normals
  - GPU K-mean algorithm
  - K is determined by studying distribution of normals
- Placement of K fill-lights along averaged normals

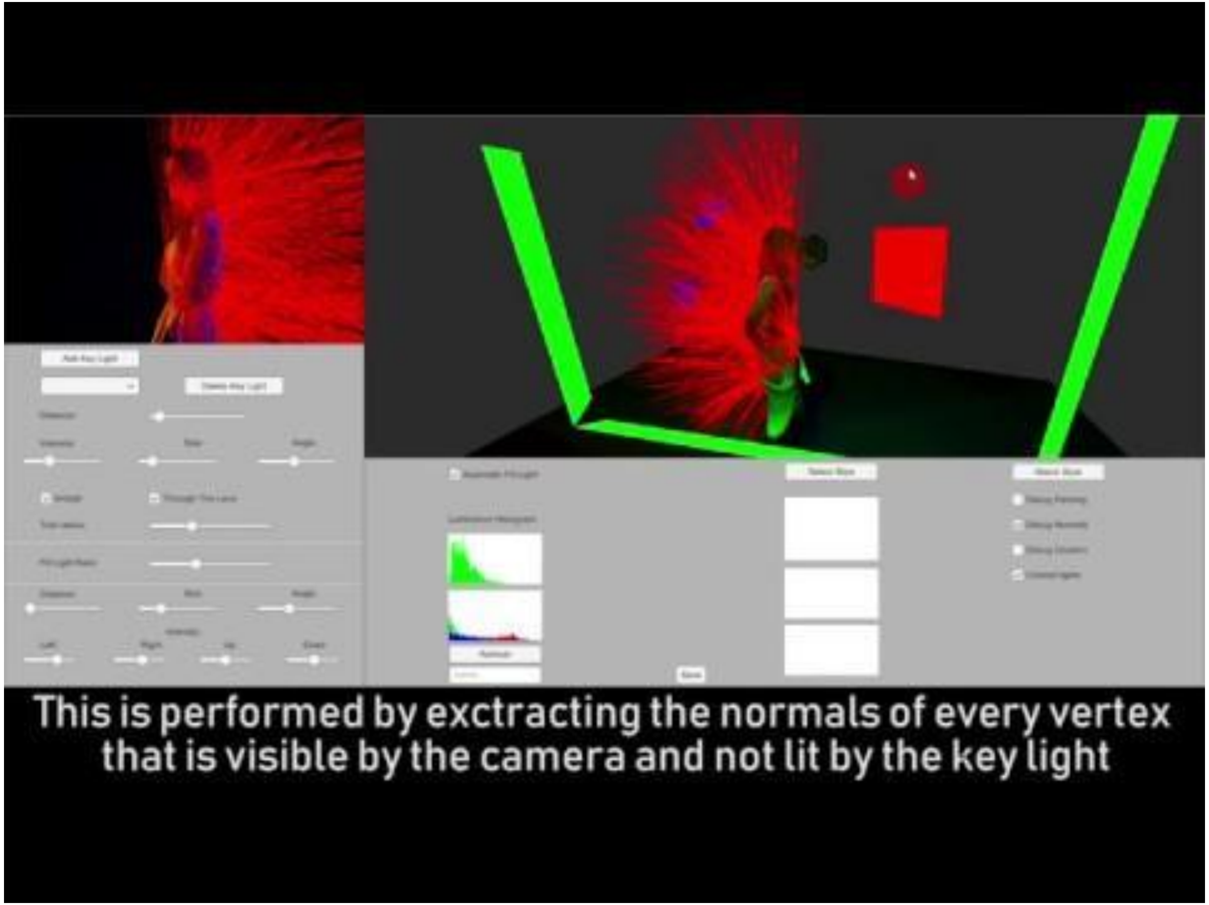




- Back lights needs to Follow some constraints
    - Cannot be visible from camera
    - Remains oriented towards the actor
    - Stays behind the subject
- We designed a rim light rig
    - Composed of 4 area lights
    - Each individually controlled in terms of intensity
    - Slides along the camera frustum



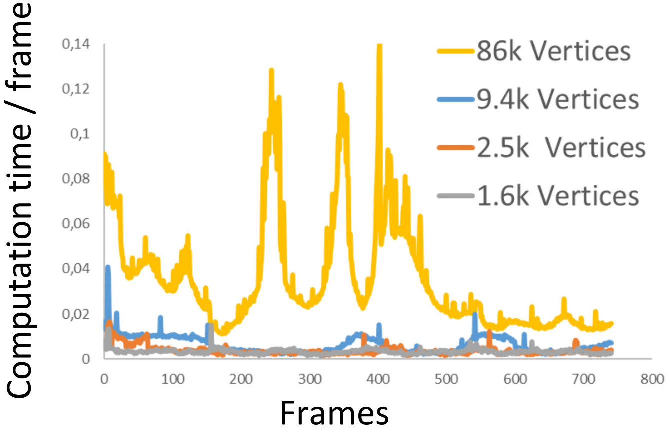
Video ● ● ● ●


This is performed by exctracting the normals of every vertex
that is visible by the camera and not lit by the key light

➢Fill-light placement method tested on a variety of 3d models with different resolutions

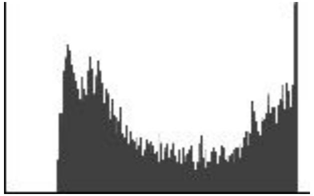
➢The technique remains real-time

➢Average of 30 fps for the highest mesh resolution



➢ How to characterize an image? By histogram of luminance

- Agnostic of any geometric information
- Histogram computed on detoured images



➢ Parameters for each light source

- flux, size, opening angle, distance, span

Light type	Optimization steps
key-light	f, s, λ
fill-light	f, s, λ
rim-light	f
all rim-lights	d, σ, λ

➢ Optimizing

- Objective function: optimizing the mutual information (distance between the histograms)
- Use of Particle Swarm Optimization (rather robust to local minima)
- Warm starting points

Gradient histogram ● ● ● ● ●

➢Luminance histogram not sufficient

- We need to account for light direction

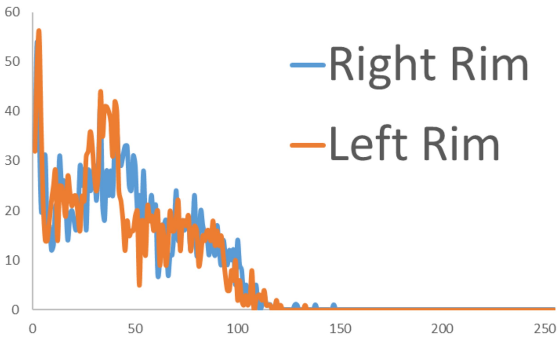
➢ Compute the gradients of the image

➢ Measure distance between histograms of gradients as well as luminance

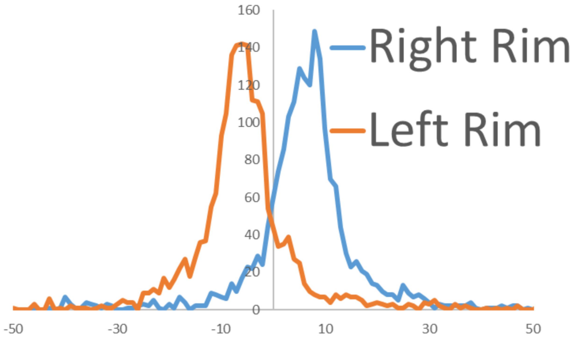
➢ Extracts the main direction of the light

Left Rim

Right Rim



Luminance histograms only



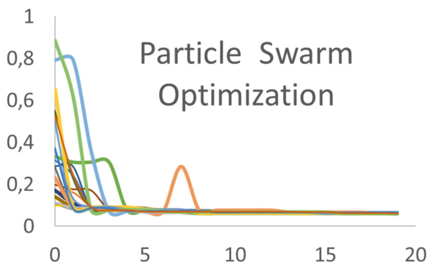
Gradient histograms only

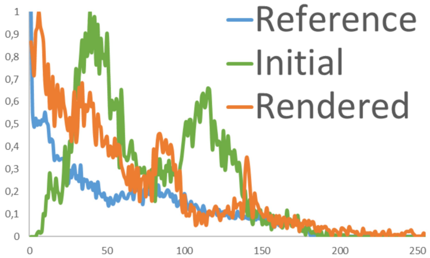
- ➢The method converges quickly
- ➢The final histograms matches the reference histograms
- ➢The light direction is properly reproduced

Reference Image

Initial Image

Result





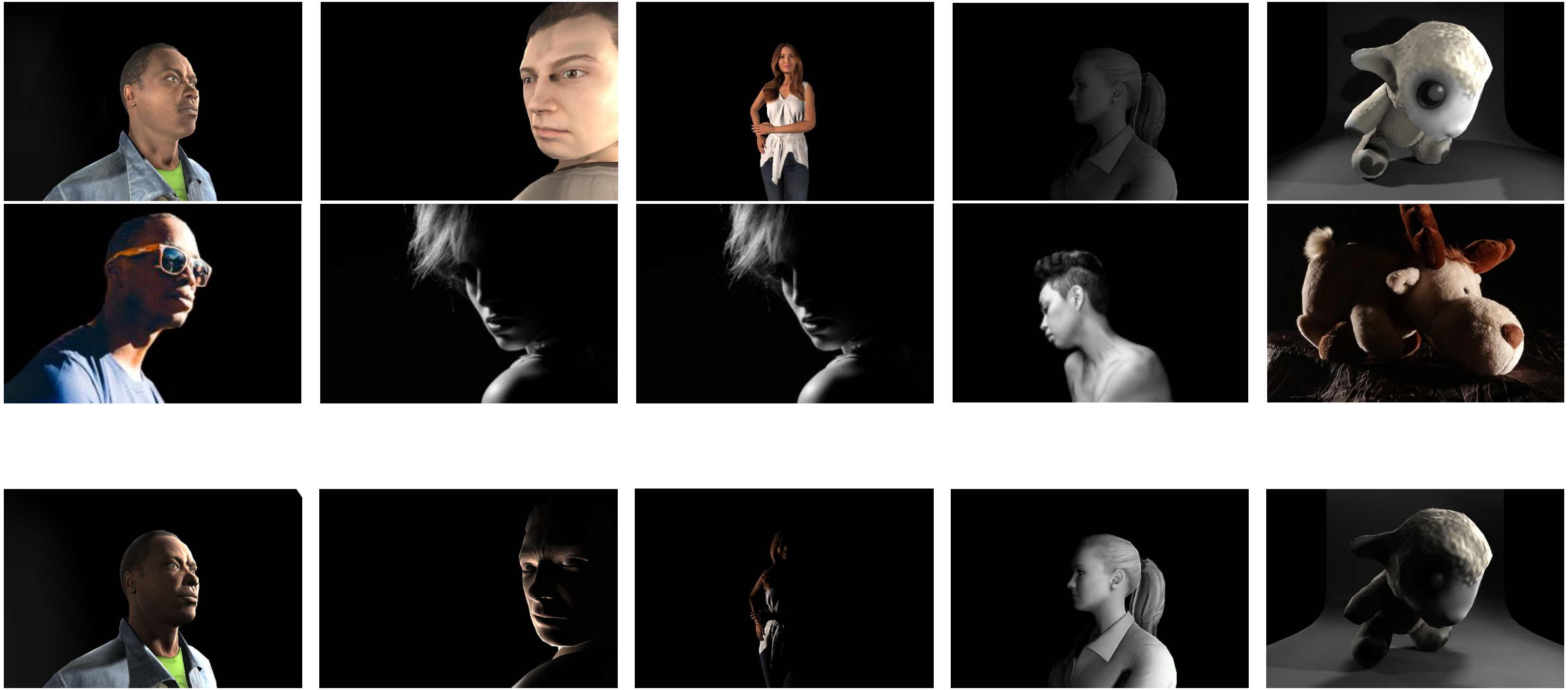
Luminance histograms

Additional results

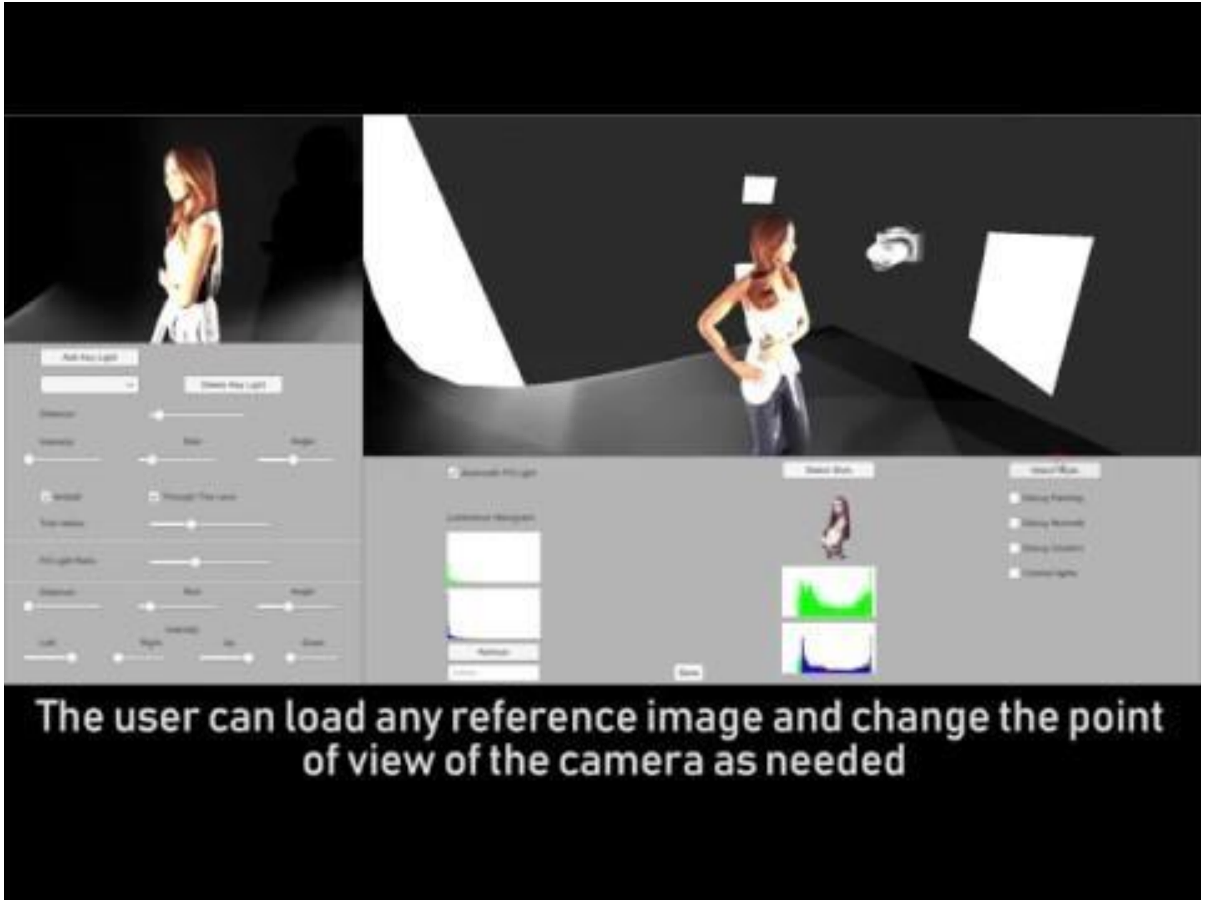
Initial lighting

Reference Image

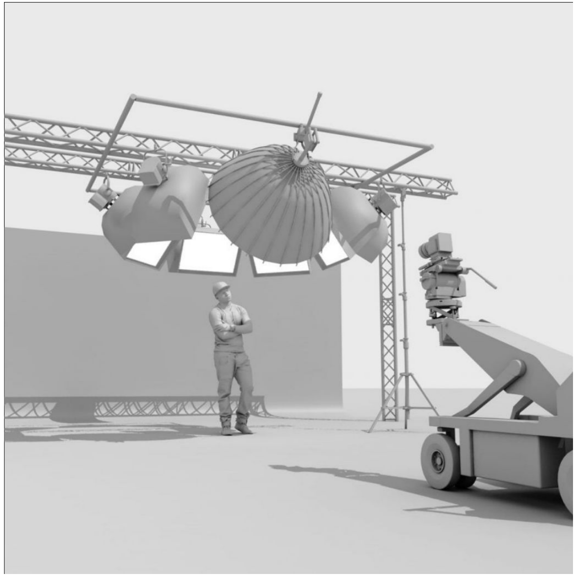
Result



Video • • • • •


The user can load any reference image and change the point of view of the camera as needed

Perspectives



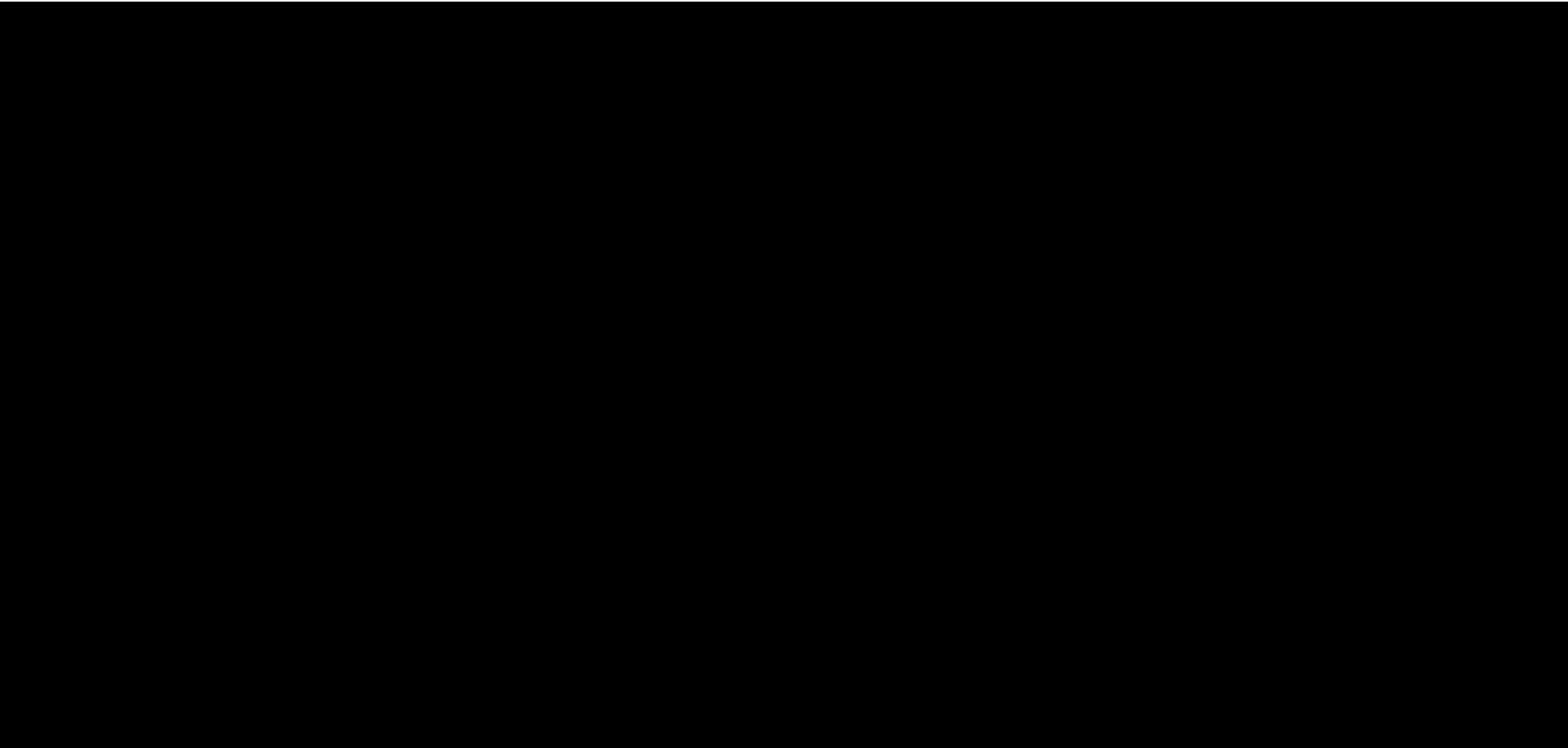
- ➢ Work on lighting the background
- ➢ Optimize for real lighting equipment
- ➢ Look into various applications
  - Fast lighting setup for previs
  - Application to AR (i.e. reproduce the real lighting)

Part 3 - Our VR Tools

VR Tools

Puppet Master

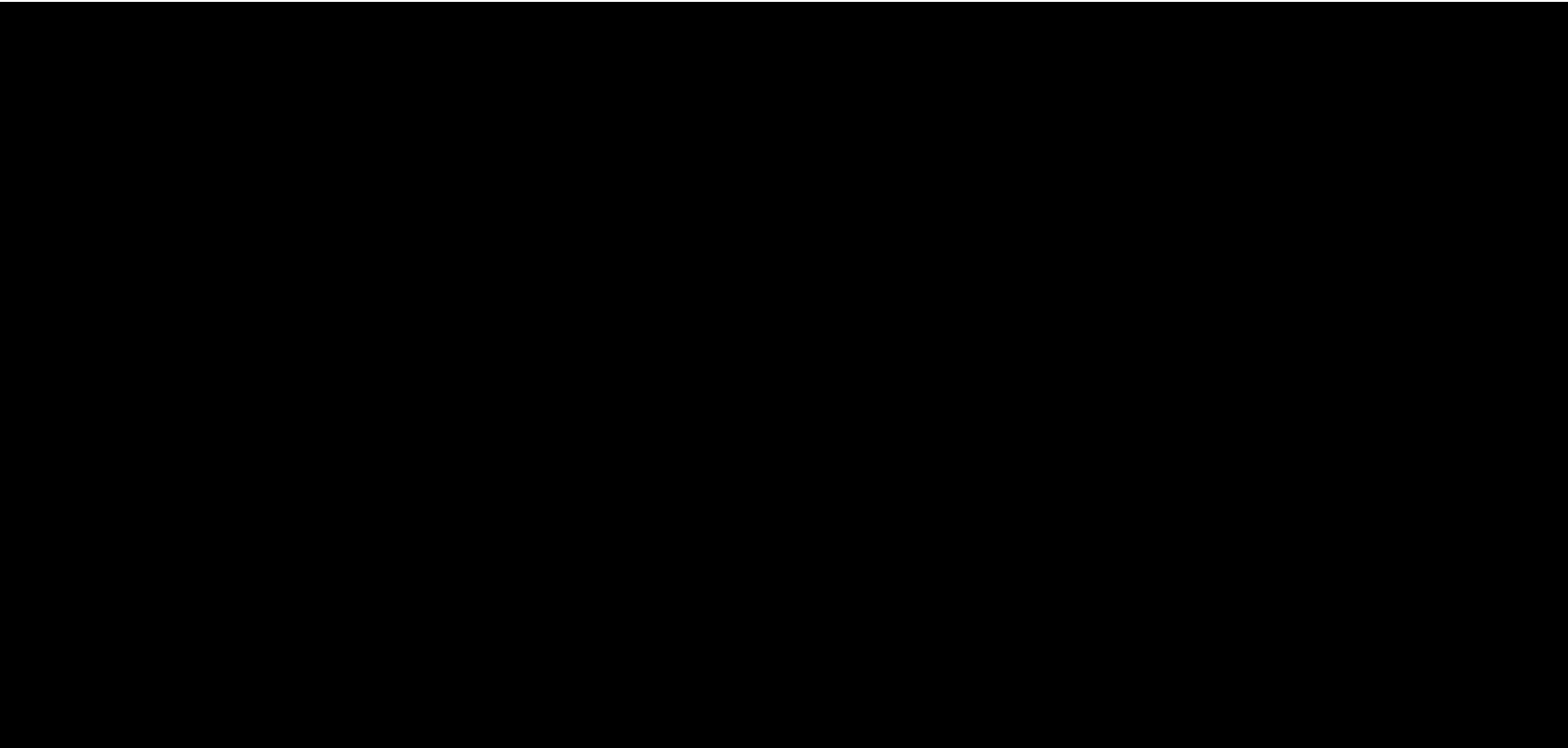
In the cut



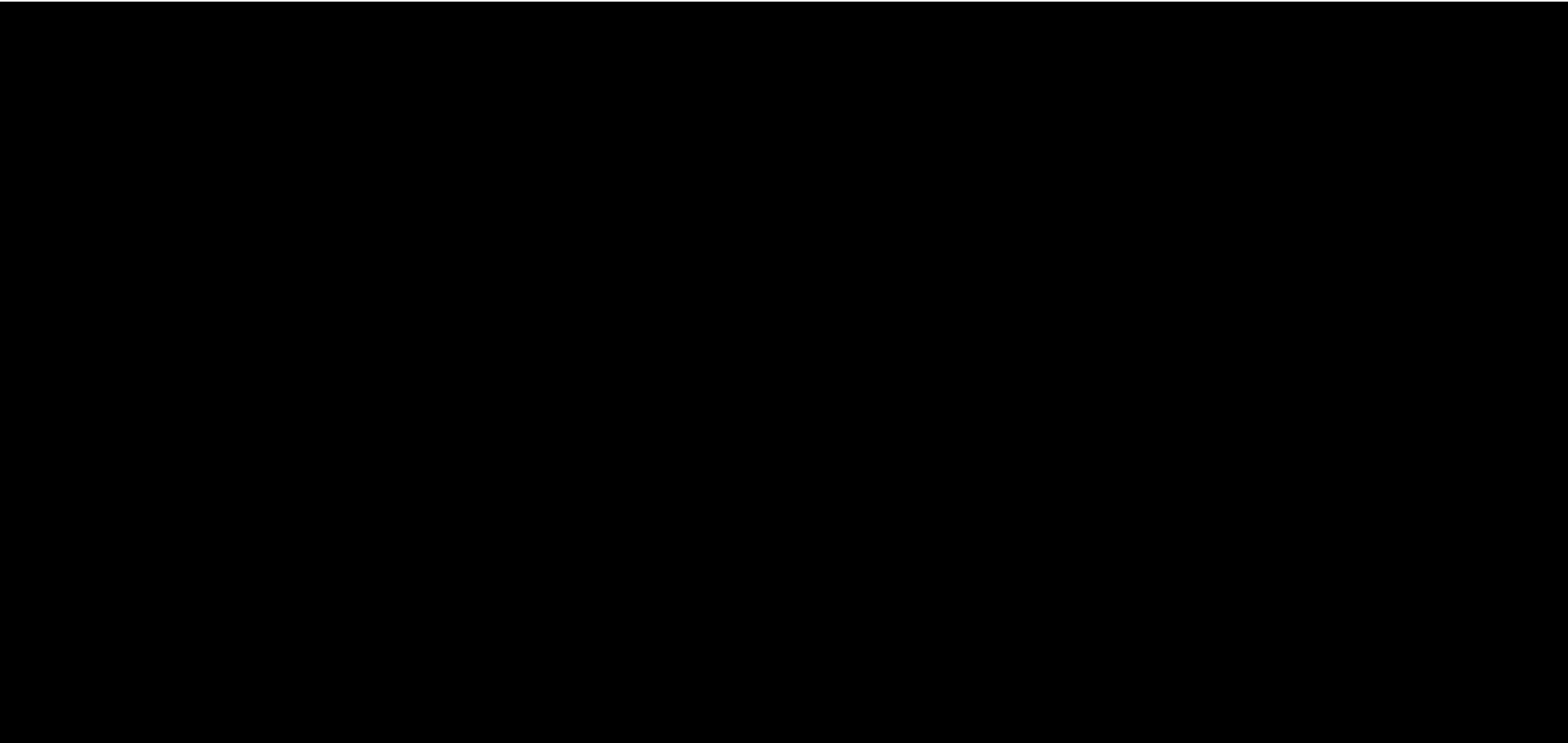
Acting out

Adding complementary cameras

Just Shoot Me!



The Railway Man



Part 4 - Our Drone Tools

Drone Cinematography

Drone Cinematography

A wide-spread technique in the past 10 years (drone film festivals)

See “The circle” movie (DJI) entirely shot by a drone. Cheap technology gives aspiring producers ability to match hollywood (see this drone)

Drone Cinematography

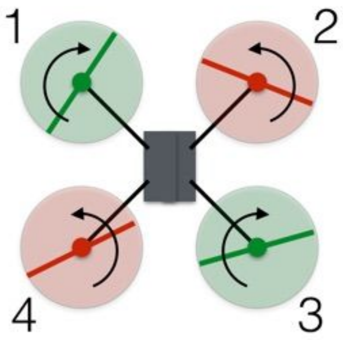
A wide-spread technique in the past 10 years (drone film festivals)

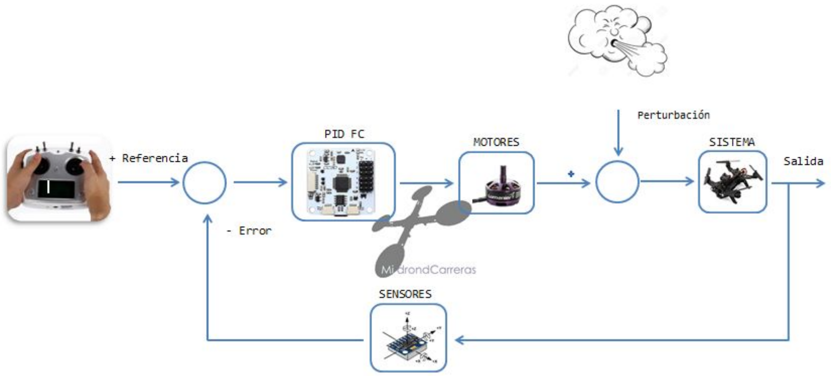
See “The circle” movie (DJI) entirely shot by a drone. Cheap technology gives aspiring producers ability to match hollywood

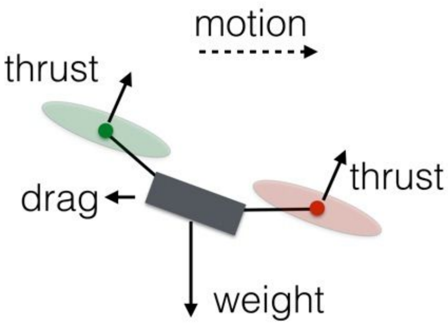
What is a Drone, How Do You Control It?

Drone = autonomous control

- Four engine speeds to regulate
- A PID controller uses the difference between a current configuration and a desired configuration to compute four speed signals







A Challenging task

No Film grammar for drones (yet) - see multidrone.eu

Generally two persons required :

- one to control drone's motion
- one to control drone's orientation (framing)

Requires skilled operators

=> very hard to synchronize with objects in motion

=> timing is essential

How Smart Are Commercial Drones Today?

- Follow-me technologies to frame a target
    - Using the GPS position of a target
    - Or uses image-based visual tracking
- Control by gestures
    - Image-based analysis (take off, approach, left, right, up)

Can we make them even smarter?

- Can they decide on optimal view angles?
- Can they compute qualitative motions?
- Can they understand cinematographic language?

SmartER Drone Cinematography

Research challenges:

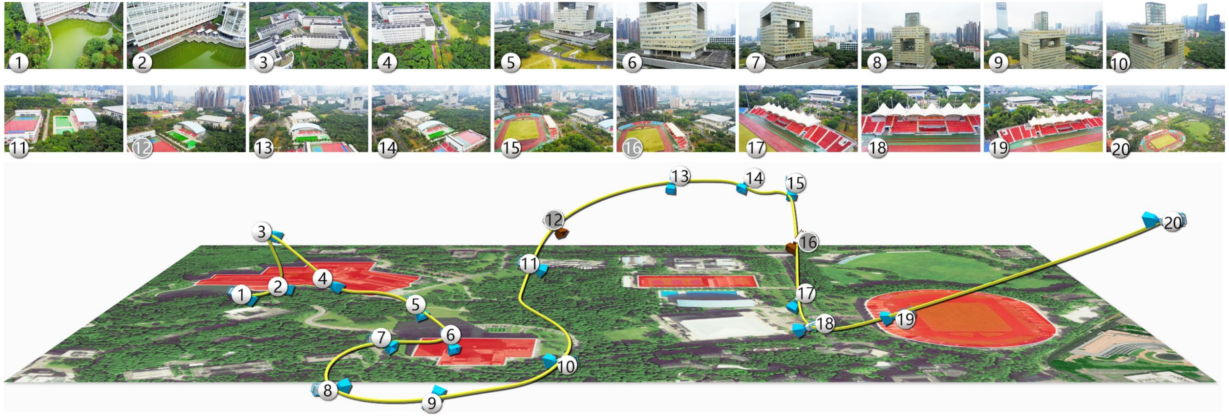
- *Formalize* film knowledge for drones
- Plan paths of *cinematographic quality* at a low computational cost
- *Ensuring safety* at all times

Automated Cinematographic Drones

Drone Videography for flybys [SIG-18]

Motivations:

➢ Generate an aesthetic flyby of given buildings



Issues:

➢ Complex tasks for novice users

➢ Requires multiple trials

➢ Generated videos are often not qualitative

Drone Videography for flybys [SIG-18]

Motivations:

- ➢ An aesthetic flyby of given buildings and their environments

User tasks:

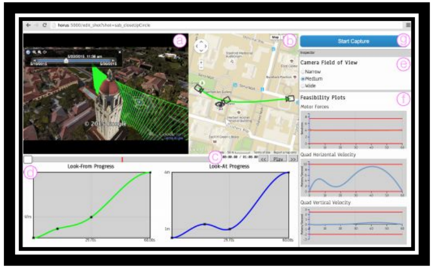
- ➢ Choose the camera angles, choose the camera motions around buildings, choose the transitions between buildings?
- ➢ Create smooth (cinematographic) trajectories
- ➢ Ensure safety (eg. when drone is hidden by a building)

Issues:

- ➢ Complex tasks for novice users
- ➢ Requires multiple trials
- ➢ Generated videos are often not qualitative (for novice users)

Existing work

Horus

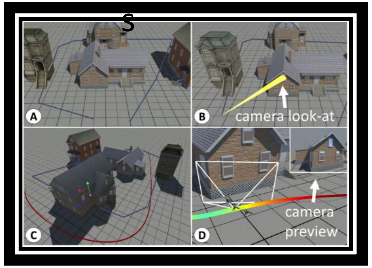


[Joubert et al., 2015]

- Intuitive Interface
- C4 trajectories
- Simulation

- Offline
- Dedicated to outdoor environments

Airway
s



[Gebhardt et al., 2016]

- Intuitive Interface
- C4 trajectories
- Obstacle Avoidance

- Offline
- Dedicated to static scenes

Drone Videography for flybys

Automating this process is computationally complex:

- ➢ How to choose the best viewpoints among an infinity of possibilities? What is a “best viewpoint”?
- ➢ How to generate best trajectories? What is a “best trajectory”
- ➢ How to plan a complex sequence of trajectories?

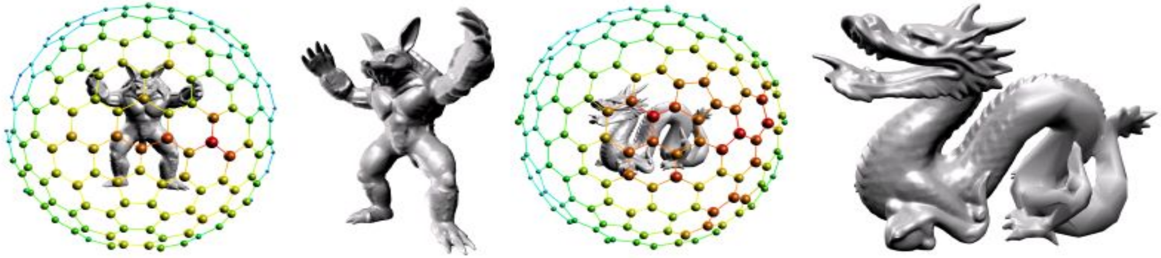
Our approach [SIGGRAPH 2018]

1. Provide a quality metric for views of buildings (called landmarks)
2. Generate qualitative **camera moves** around landmarks
3. Connect the different camera moves

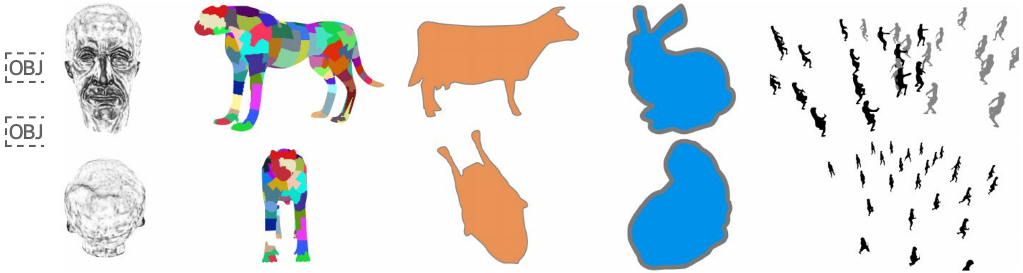
Viewpoint quality

Viewpoint entropy [Vasquez’01]

- ➢ Defines how much information a viewpoint conveys

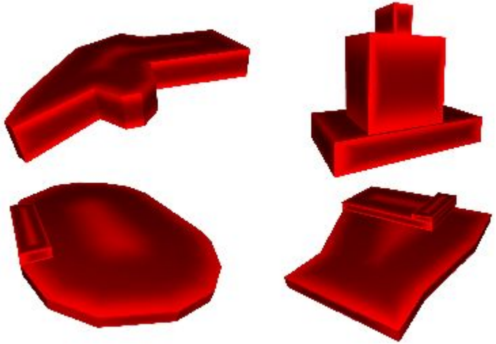


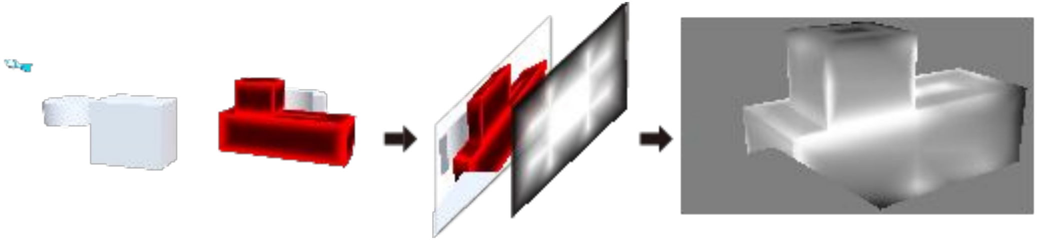
- ➢ Different critera (mean curvature, visibility, alignment, silhouette complexity, visual dispersion)



Viewpoint quality

1. Compute saliency of buildings:
   - ➢ Edges provide information on the shape
   - ➢ Centers of areas
2. Compute a “line of thirds” overlay
   - ➢ Regions in the center are preferred
   - ➢ Regions along the ⅓ axes are preferred
3. Compute both information
   - ➢ Results in a viewpoint quality (sum of information)

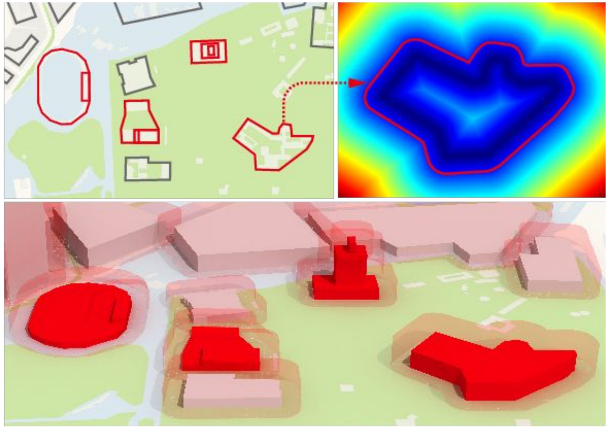




Ensuring safety

Expand 3D buildings with a safety area

- using a surface Minkowski sum (sphere)



Composing Viewpoint quality

- creates a scalar field through the scene

Creating camera moves (1)

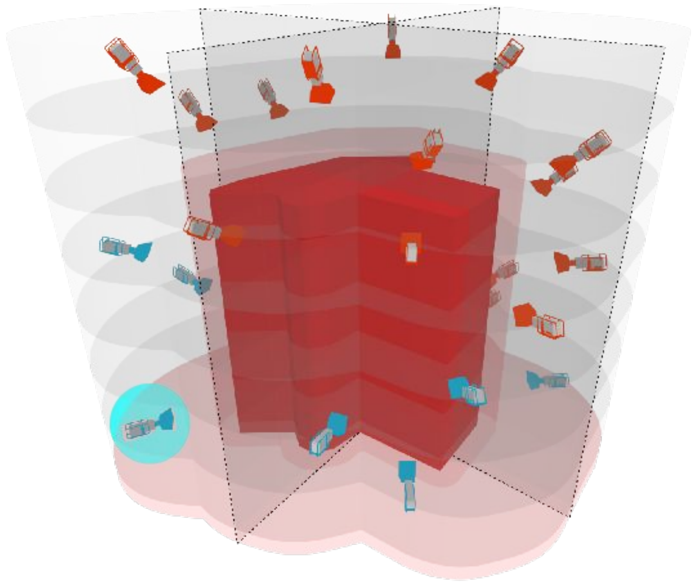
How to create *interesting moves* around a building?

- ➢ Should have a minimum change in height or in angle
- ➢ Should connect good viewpoints

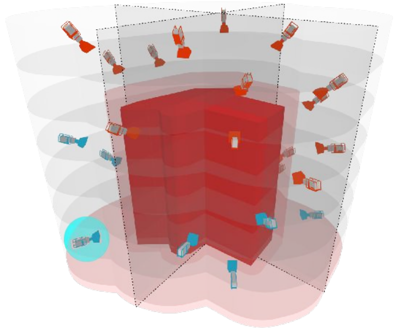
We propose spatial partitions around each building

- ➢ Horizontal partitions (max 7 layers)
- ➢ Vertical partitions (4 partitions)

The best viewpoint is computed in each partition



Creating camera moves (2)



A subset of all possible moves is created

- ➢ Only create moves across a minimum of 4 partitions (horizontally + vertically)

Each trajectory is evaluated (192 possibilities)

- ➢ Quality of the viewpoints along the move

A selection of the *n* best moves is performed

Chaining camera moves

Scene is composed of *m* landmarks

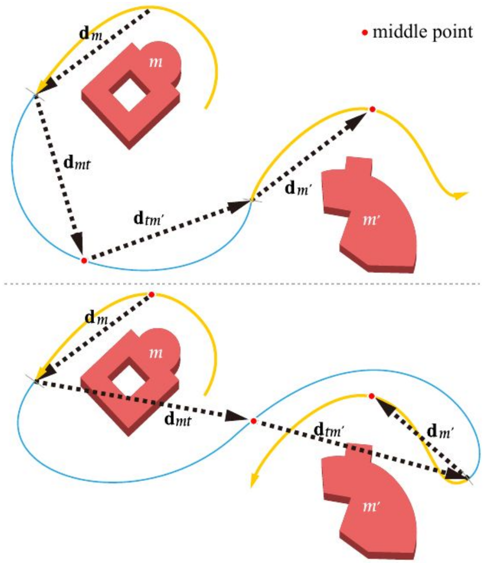
For each landmark: *n* best moves

How to compute an optimal trajectory?

- Generate all transitions between possible moves
- Evaluate the quality of each transition
  - ➢ Length, curvature, change of directions

Now each move has a quality (cost), each transition has a quality (cost), we search for the shorted path through landmarks

=> looks like a Travel Salesman Problem



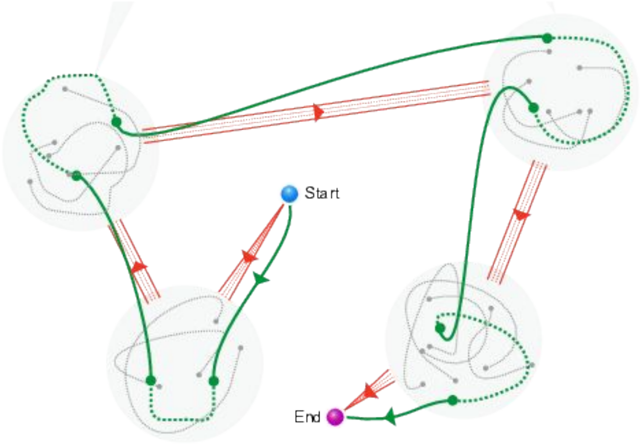
Solving: Set-TSP

A specific case of the TSP:

- we only need to visit ONE move per landmark
- corresponds precisely the Set TSP (or one-of-a-set TSP) [Noon93]

Table 1. Test scene statistics: number of landmarks ($\#m$), the total time for computing view quality fields, local trajectory construction time, global optimization time, and distance of the global optimal trajectory in meters.

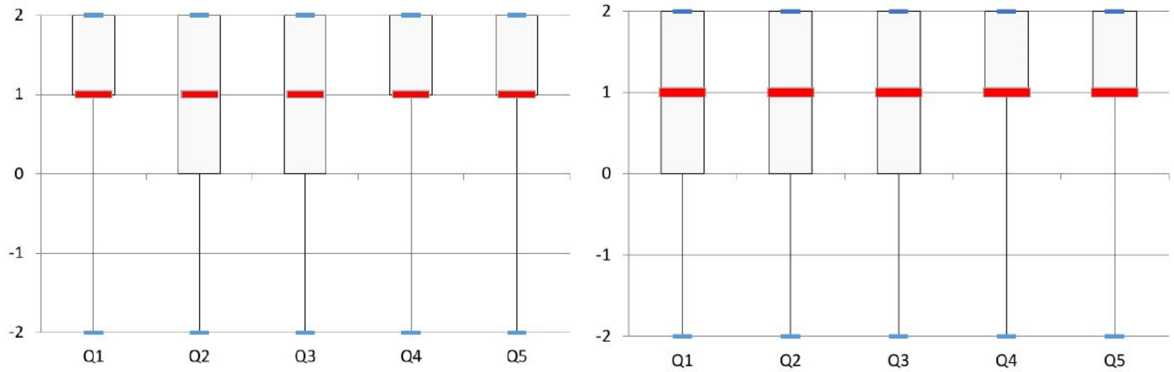
Figure	$\#m$	Time$_f$	Time$_l$	Time$_g$	Distance
Fig. 1	3	5m	15s	10s	2475
Fig. 2	4	7m	37s	15s	2179
Fig. 12	3	5m	18s	15s	3190
Fig. 13	4	10m	41s	38s	3806
Fig. 14	5	8m	50s	31s	2998



User evaluations

Compared three videos. Given the same landmarks:

- Auto: using our automated approach
- Manual: manually flying the drone to shoot the landmarks
- DGS: use DJI GS Pro software on iPad to design a drone path, and run it



(a) *Auto* vs. *Manual*

(b) *Auto* vs. *DGS*

Q1; more pleasing video

Q2:clearer overvies of landmarks

Q3:follows a more reasonable route

Q4:provide better transitions

Q5:create smoother trajectories

Results

The computed trajectories are sampled and sent as a sequence of GPS waypoints to the drone (DJI Phontom 3 Pro)

Reactive Cinematographic Drones

Joint work with technicolor

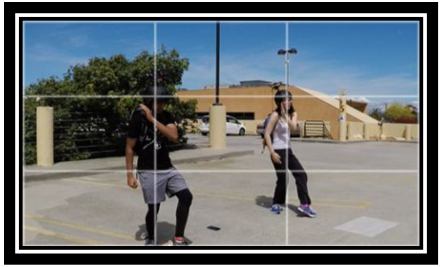
Motivations

- Have drones that can frame and “understand cinematographic language”
  - On angles: Over The Shoulder shot, Apex shot,
  - On sizes: Medium shot size
  - On framing: placement of targets on the screen

- Have drones that maintain cinematographic properties
  - Adapt to changes in the scene (actors locations and orientations)
  - Ensure cinematographic quality in camera motions

Existing work

Shot design

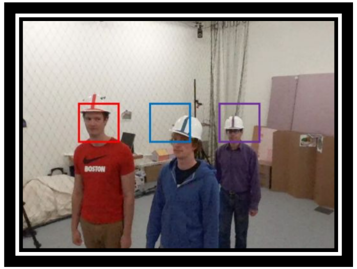


[Joubert et al., 2016]

- Based on the Toric Space
- Maintain a given framing

- No obstacle avoidance
- No visibility checking
- Limited motion of actors

Framing based control



[Nageli et al., 2016]

- Realtime
- Obstacle Avoidance
- Frame more than 2 actors

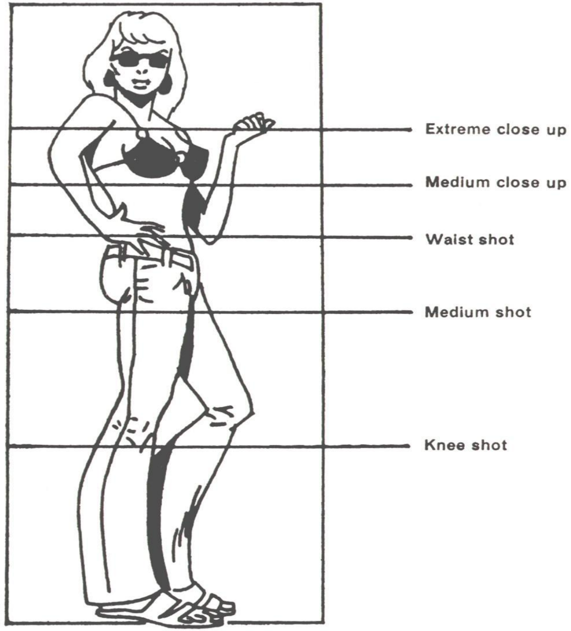
- Limited interactions
- Non-cinematographic paths

How to Frame with a Drone?

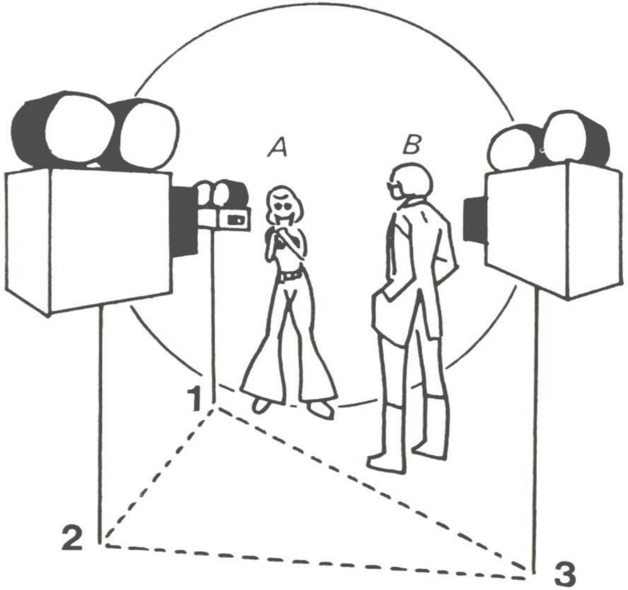
From Cinematographic Properties to Viewpoints

Cinematography

An empirical language defined by cinematographers:



Shot size



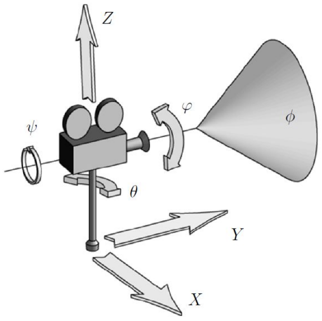
Shot angles: 1+3 OTS, 2 Apex

Composition

From Properties to Viewpoints

From film language to camera viewpoints:

- A computationally complex problem addressed with optimization
- Each visual property is defined as a cost function on camera parameters (7 dofs)



- All visual properties are aggregated in a cost function

$$F(c) = \sum_i f_i(c)$$

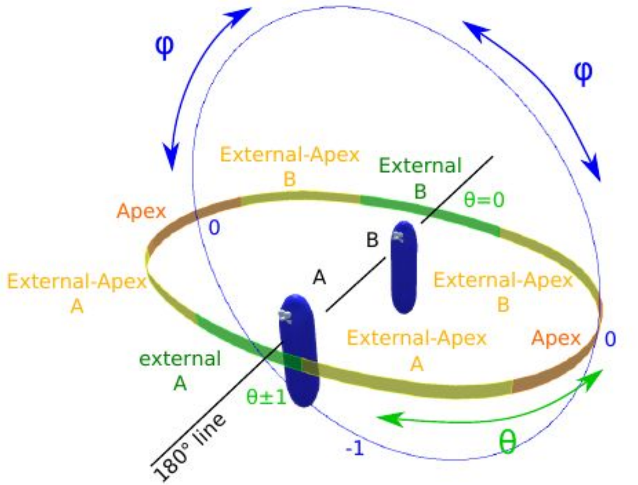
- A non-linear solver searches for best viewpoints
- Computationally expensive (stochastic solvers [Ranon15])

=> we propose a novel parametric space for camera composition problems

The Toric space

Enables an algebraic expression of cinematographic properties:

- Screen composition
- Horizontal and vertical angles (theta, phi)
- Distance to targets

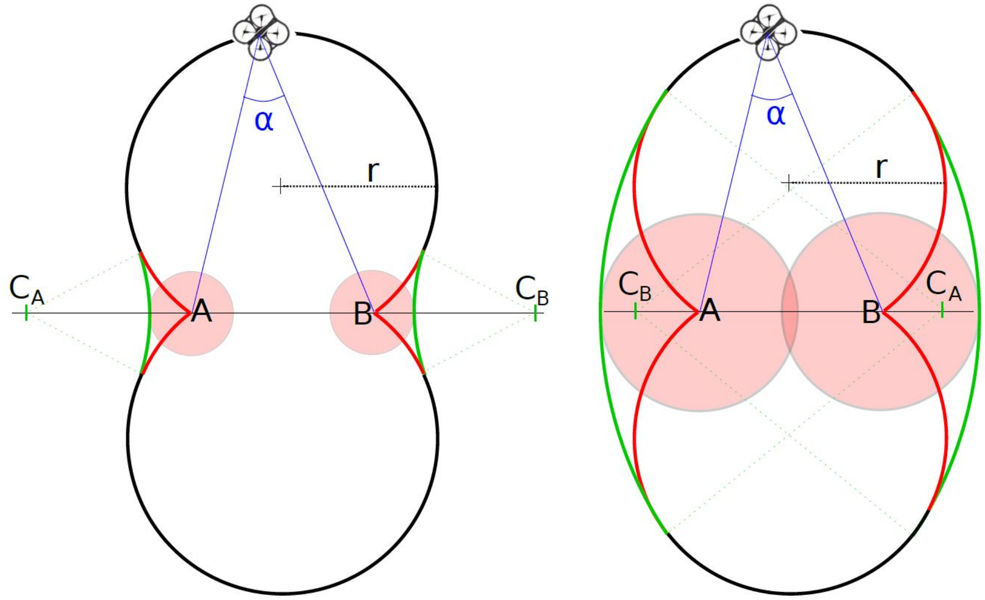


Cameras can therefore be controlled in an algebraic way

=> casts a 7D camera problem into a 3D camera problem

The Drone Toric space

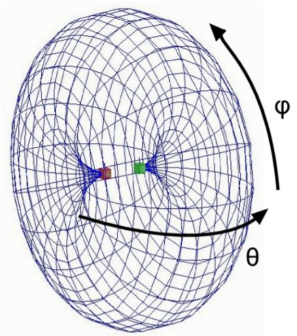
- ➢Adapt the Toric space to drones
  - ➢ To ensure actors’ safety (targets A and B)

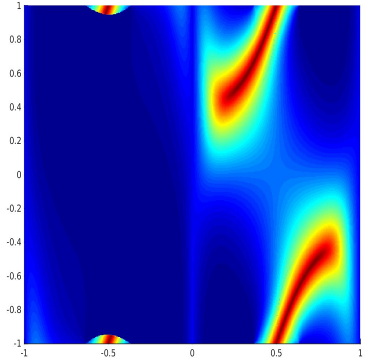


Type #1 Type #2

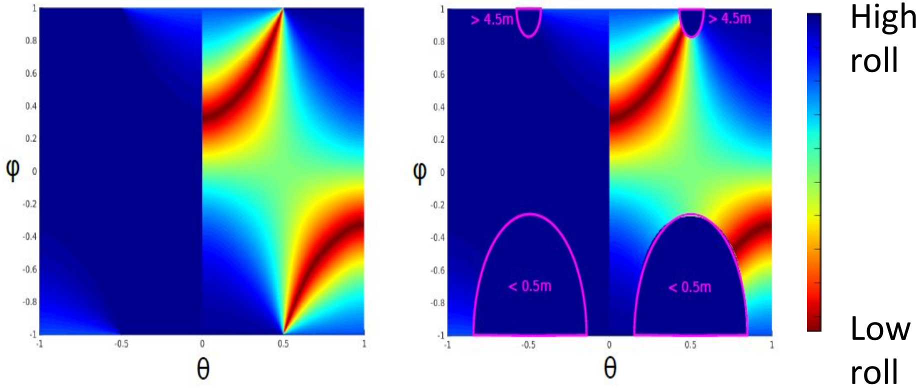
Image-space Interaction

- ➢Additional interactions
- ➢Better optimization scheme
  - ➢Use the roll as cost function
  - ➢Account for obstacles





Cost function
[Lino et al. 2015]

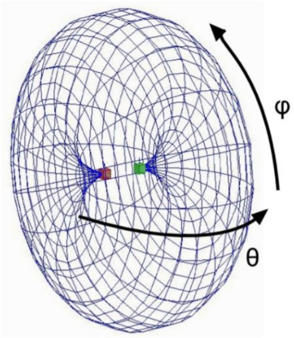


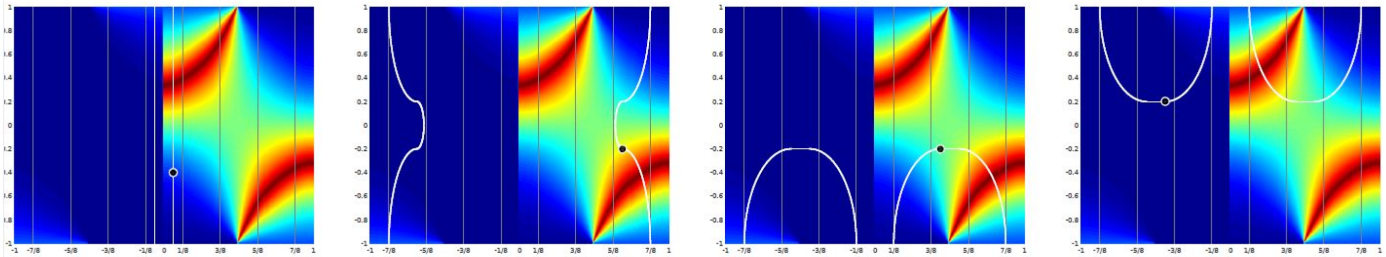
Our cost function

With Obstacles

Image-space Interaction

- ➢Additional interactions
- ➢Better optimization scheme
  - ➢Use the roll as cost function
  - ➢Account for the obstacles
  - ➢Adapt the search to the current position



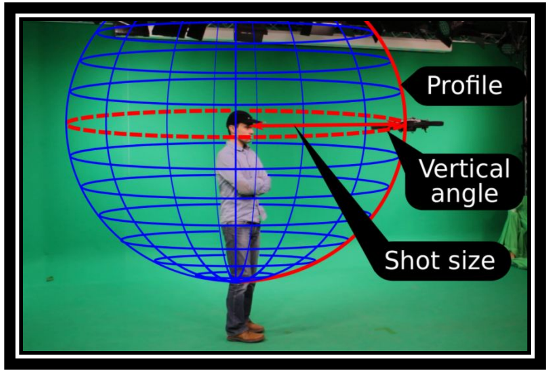


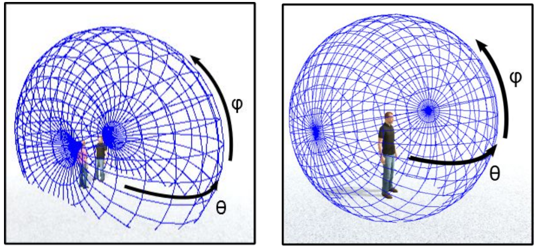
(a) External A (b) External-Apex B (c) Apex (from below) (d) Apex (from above) crossing 180° line

How to Move a Drone In a Cinematographic Way?

Creating Cinematographic Trajectories

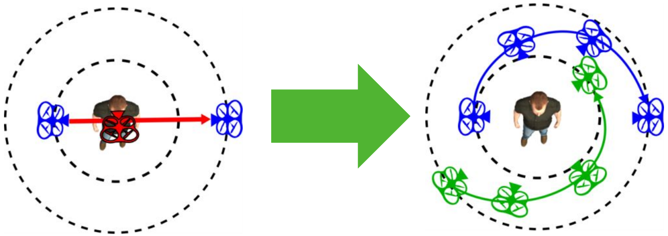
User input



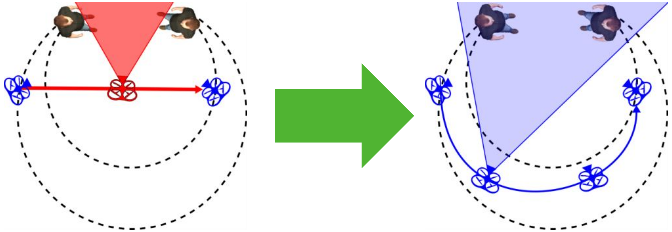


Interpolation in the Toric Space

1 actor:

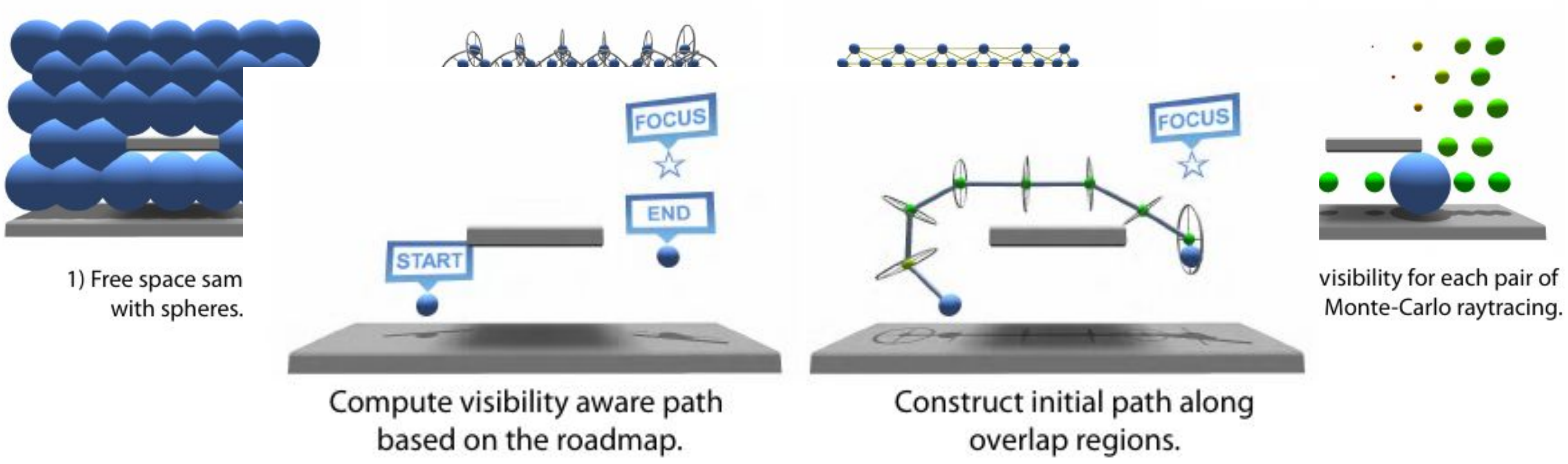


2 actors:



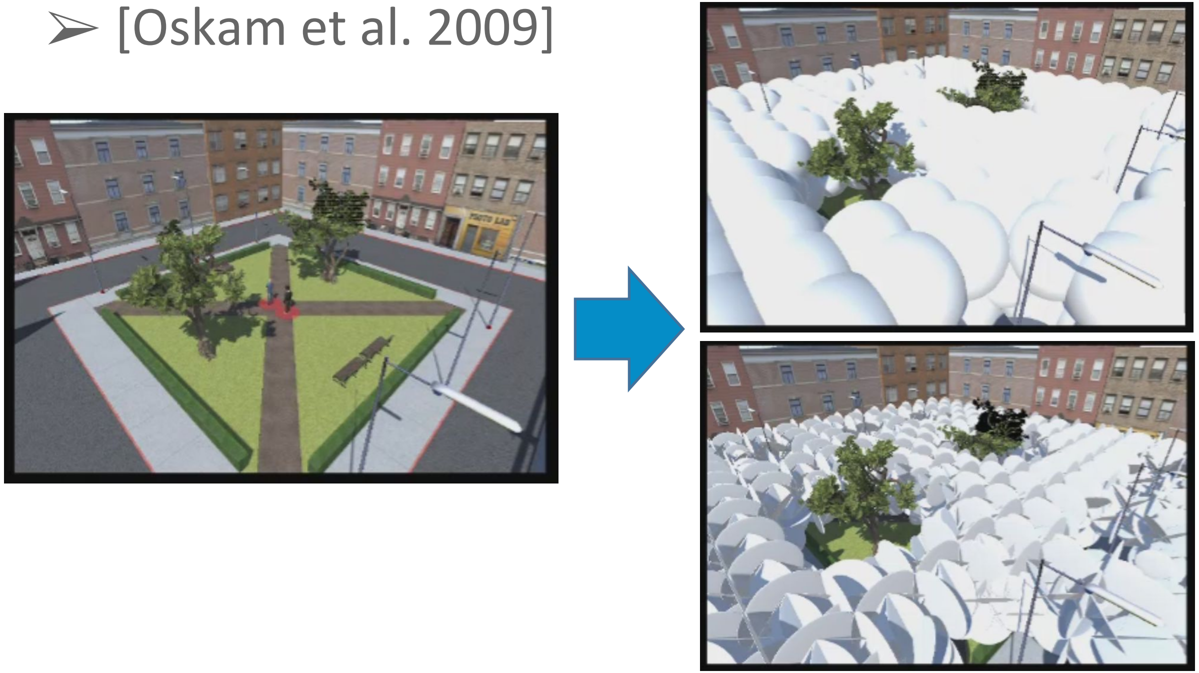
Planning cinematographic paths

- ➢Collision avoidance mandatory
- ➢Visibility aware roadmap and A* path planning
  - ➢ [Oskam et al. 2009]



Planning cinematographic paths

- ➢Collision avoidance mandatory
- ➢Visibility aware roadmap and A* path planning
  - ➢ [Oskam et al. 2009]



Planning cinematographic paths

➢Planning the path in the space of visual properties

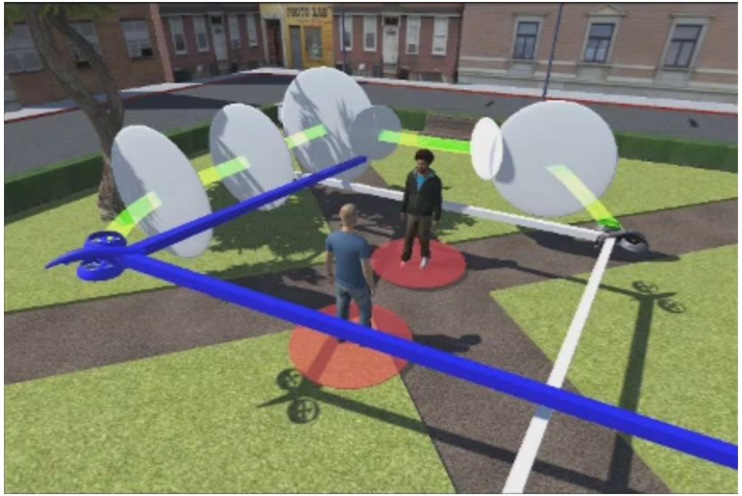
- New distance metric based on the toric space

$$\boxed{D_s^2(n_i,n_j) = d(\alpha_i,\alpha_j)^2 + d(\varphi_i,\varphi_j)^2 + d(\theta_i,\theta_j)^2}$$

- Weighted with visibility information

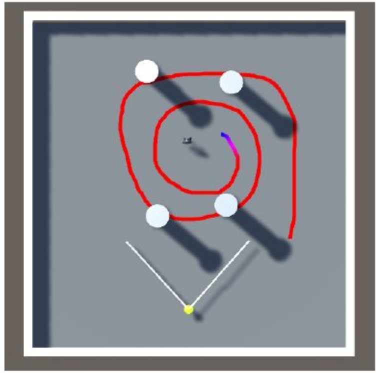
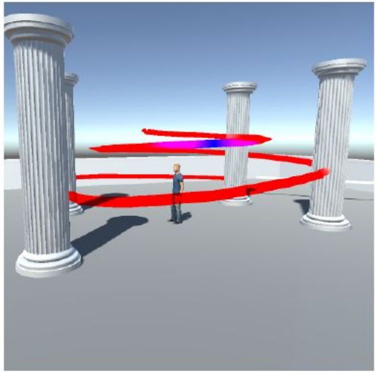
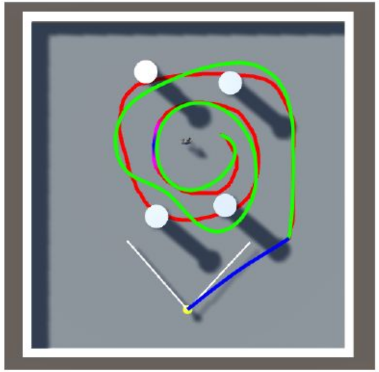
Initial Shot

Desired Shot

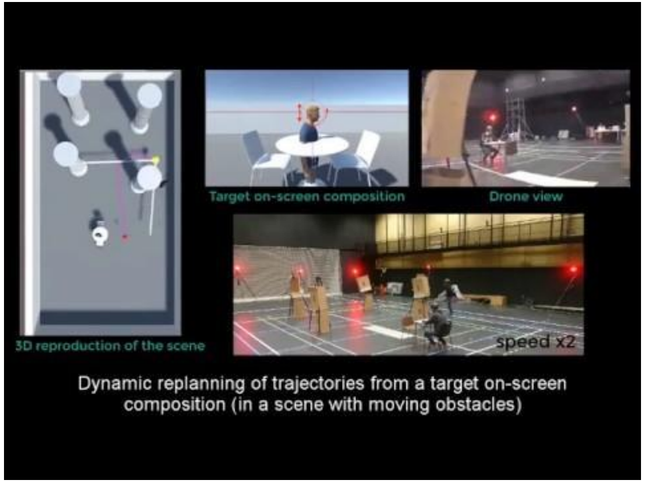


Sketching trajectories

- ➢Collision avoidance mandatory
  - Use the roadmap
- ➢Modified A* algorithm to allow loops
- ➢C4 optimization

- RESULTS
  - Indoor tracking using optoelectronic system (VICON)
  - With Parrot ARDrones
  - With Parrot Bebop2



How to Handle Multiple Drones?

Orchestration of drones

Handling multiple drones?

- How to use our technology to synchronize multiple drones?
  - Every drone covers a different angle of the scene
  - Drones offer complementary views (for further editing)
  - Drones react to changes and avoid conflicts

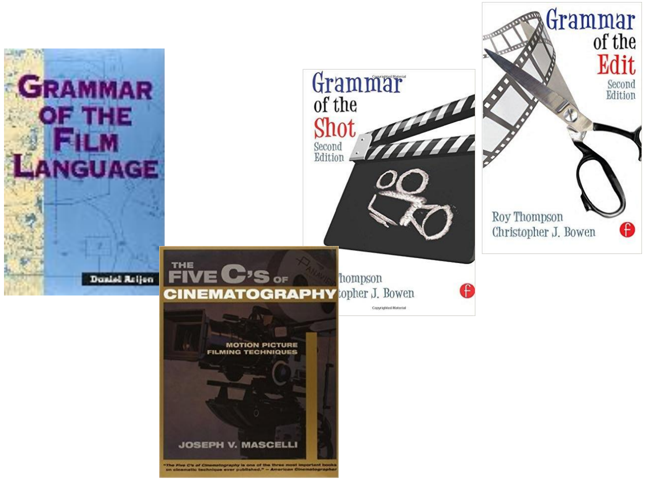
- Our approach (a TV editor metaphor)
  - A master drone (interactively controlled by the user)
  - Slave drones offering non-conflicting views that satisfy "continuity editing"

Editing

- Editing is the art of cutting between view angles
  - Choosing when to cut
  - Choosing where to cut to
  - With which type of transition

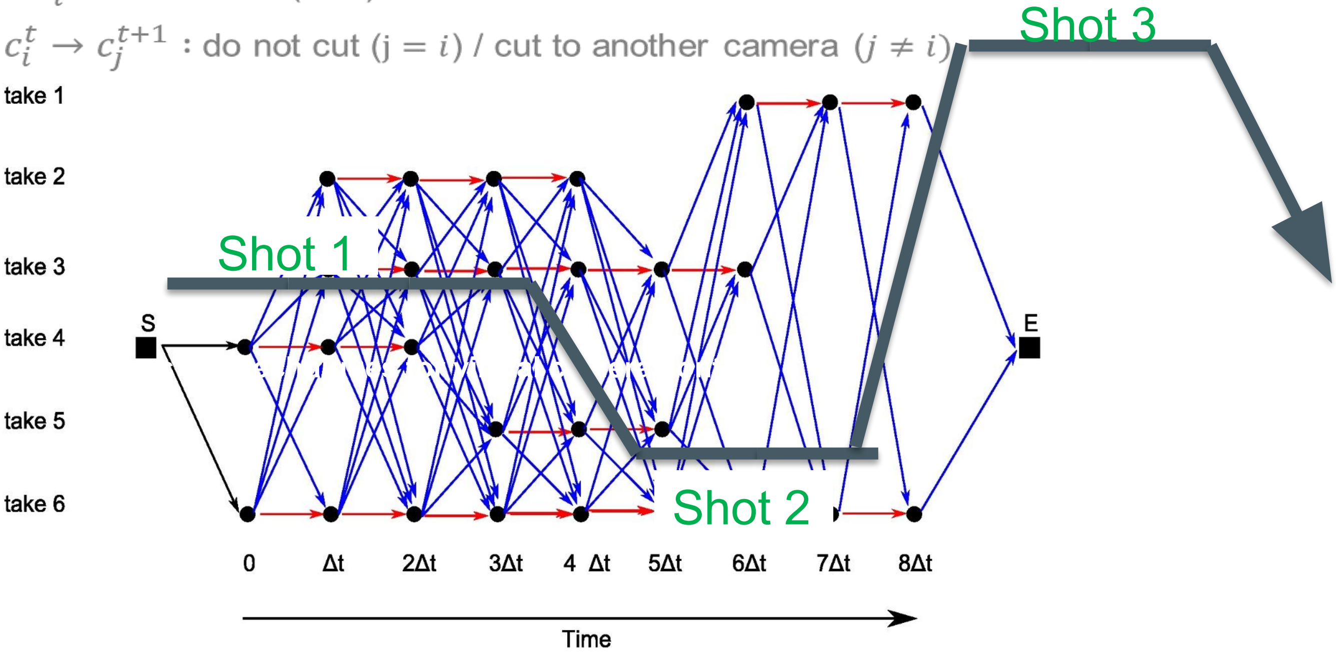
- **Editing forms a visual « grammar »**
  - Frames are letters, shots are the words
  - Scenes are sentences, films are stories

- **Continuity-editing**
  - *Grammar of the Film Language* [Arijon 76]
  - *Grammar of the Shots* [Thomson 98]
  - *Grammar of the Edit* [Thomson 93]



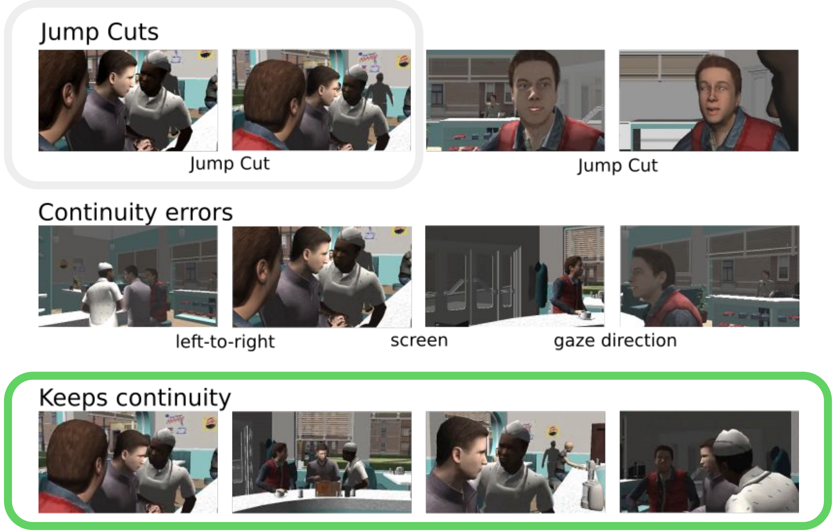
A general approach: The *"editing graph"* [Galvane etal 2015]

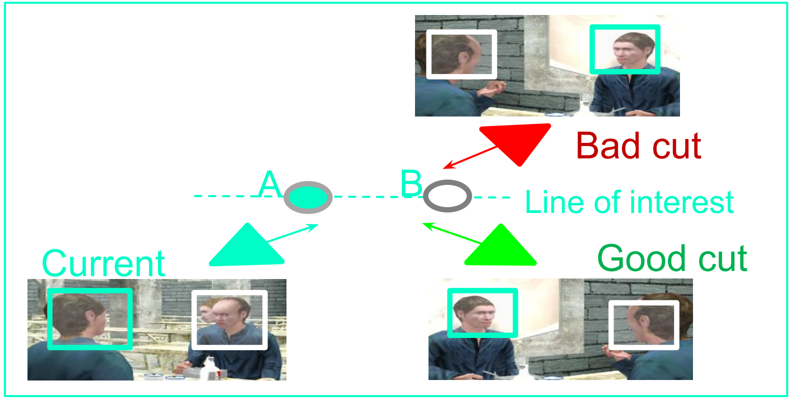
- Automated editing can be viewed as planning a path through an oriented graph
  - Node c_i^t: use camera (take) i at time t
  - Arc $c_i^t \rightarrow c_j^{t+1}$: do not cut (j = i) / cut to another camera ($j \neq i$)



« continuity editing»

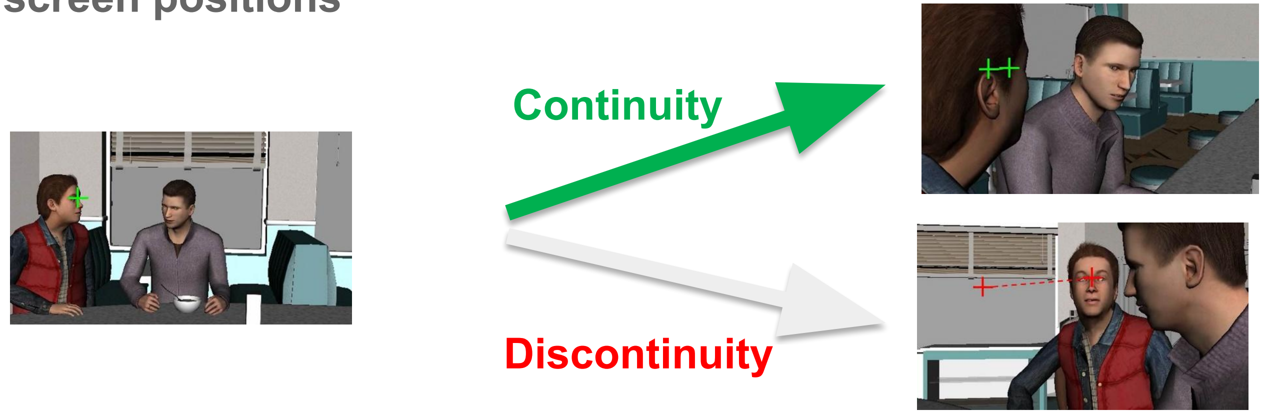
- Controls how storyline actions are perceived all together
  - **Make link** between pieces of information
  - **Guide** viewers' attention (visual cues)
- Controls how a given action is perceived as continuous in time
  - **Do not break continuity** (coherency)





Cut quality: absolute screen positions

- Penalize cuts breaking continuity
  - On **absolute screen positions**

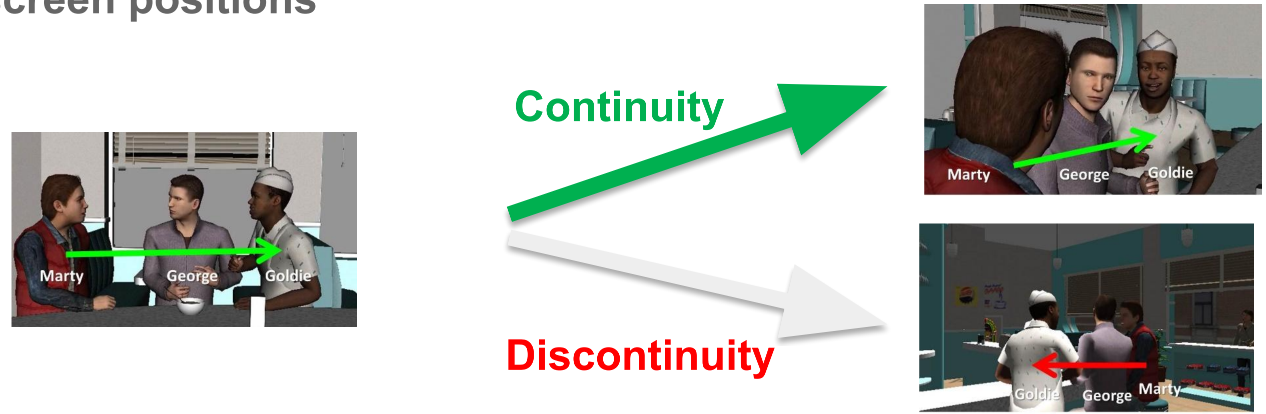


- Cost function:

$$P^{T}_{Screen}(c^t_{j-1}, c^t_j) = \sum_i \phi_S[Pos(T^i, c^t_{j-1}) - Pos(T^i, c^t_j)]$$

Cut quality: relative screen positions

- Penalize cuts breaking continuity
  - On **relative screen positions**

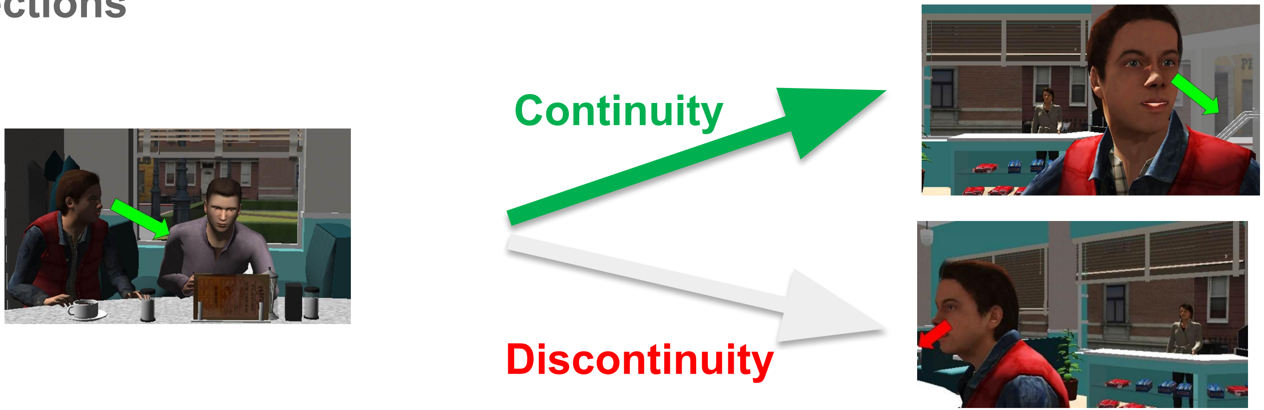


- Cost function:

$$P^{T}_{Order}(c^{t}_{j-1}, c^{t}_{j}) = \sum_{i,j} \phi_O\left[Order(T^{i}, T^{j}, c^{t}_{j-1}), Order(T^{i}, T^{j}, c^{t}_{j})\right]$$

Cut quality: gaze continuity

- Penalize cuts breaking continuity
  - On **gaze directions**

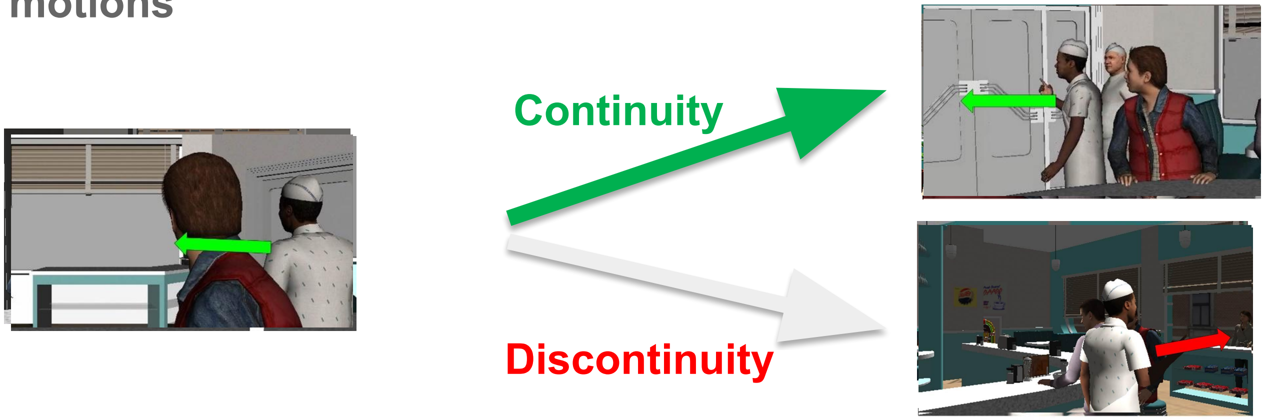


- Cost function:

$$P^{T}_{Gaze}\left(c^{t}_{j-1}, c^{t}_{j}\right) = \sum_{i} \phi_{G}\left[Gaze\left(T^{i}, c^{t}_{j-1}\right), Gaze\left(T^{i}, c^{t}_{j}\right)\right]$$

Cut quality: motion continuity

- Penalize cuts breaking continuity
  - On **apparent motions**



- Cost function:

$$P^{T}_{Motion}(c^{t}_{j-1}, c^{t}_{j}) = \sum_{i} \phi_{M}[Motion(T^{i}, c^{t}_{j-1}), Motion(T^{i}, c^{t}_{j})]$$

Avoid *“jump cuts”*

- Penalize cuts that do not look like cuts (visually, not enough **change in size or view angle**)

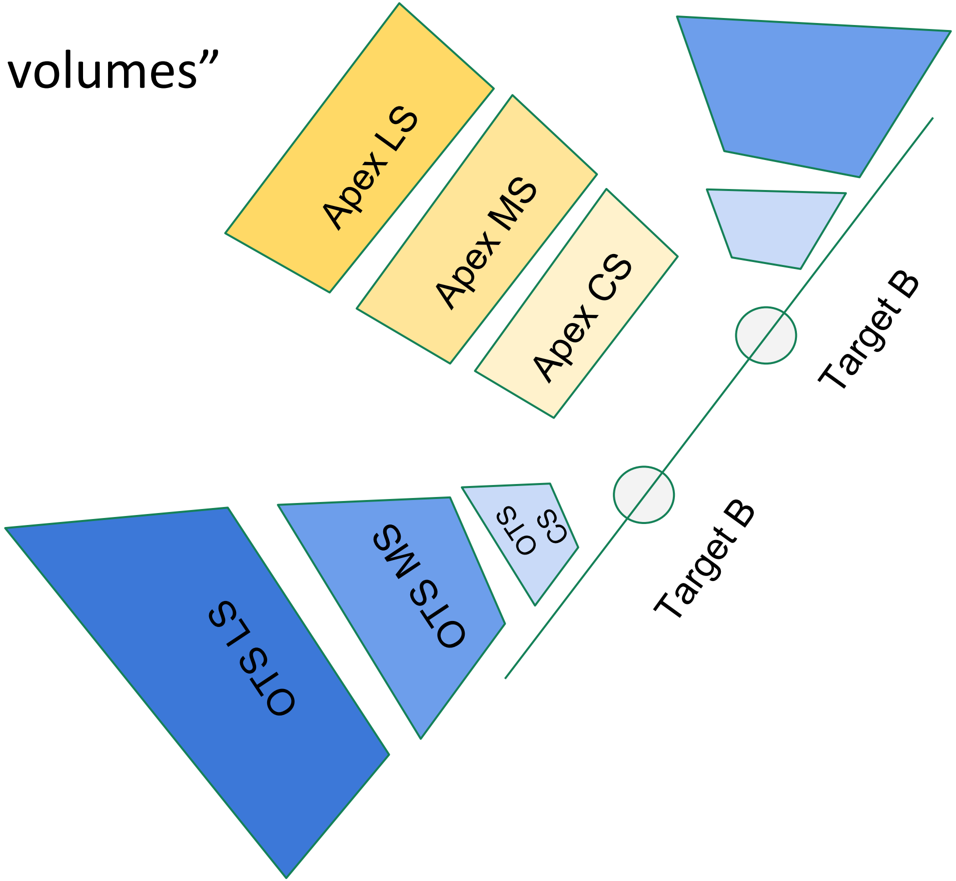
Sufficient change in size

Sufficient change in view angle

« Jump cut »

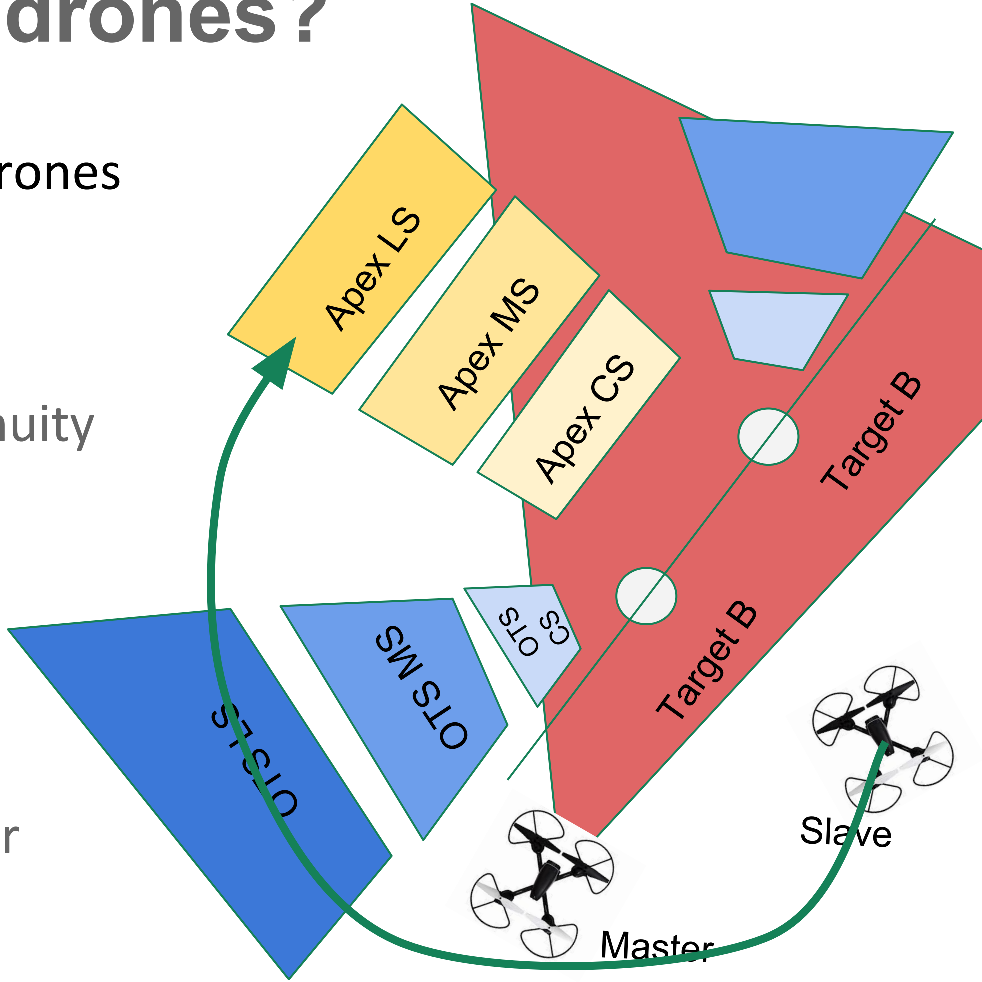
Handling multiple drones

- ➢ Define tagged regions " 18 semantic volumes"
  - In Toric space coordinates
  - Relative to the targets



Handling multiple drones?

- ➢Remove conflicting areas for slave drones
  - Remove areas with visibility conflicts
  - Remove areas that fail “continuity editing”
- ➢Select a possible volume
  - Shortest path to a volume
  - That avoids visibility by Master

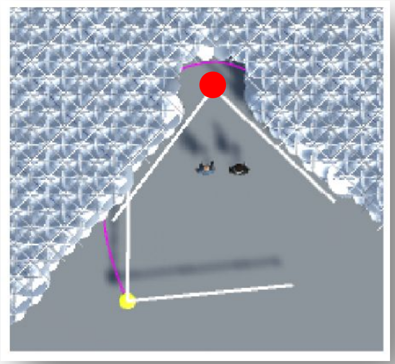


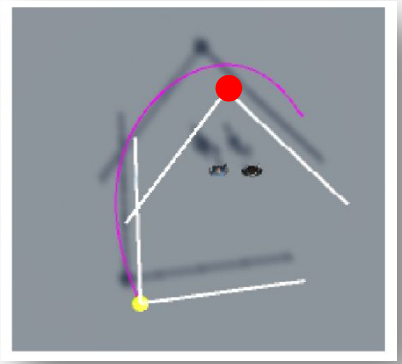
Searching for non-conflicting assignments

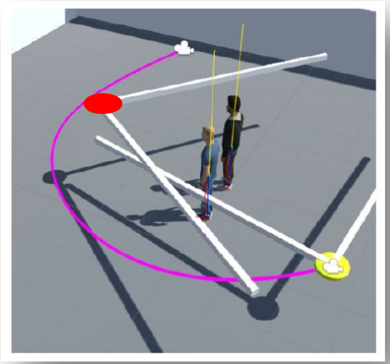
- Use a min-conflict solving process
  - Find the slave drone with the minimum number of conflicts
  - Search a semantic assignment for that drone
  - If failure, search for an assignment for the two slaves drones with minimum number of conflicts
- Practical complexity is low (even with 3 slaves)
  - 4k combinations
  - Above 4 drones, the environment gets cluttered

➢Handling planning through the roadmap

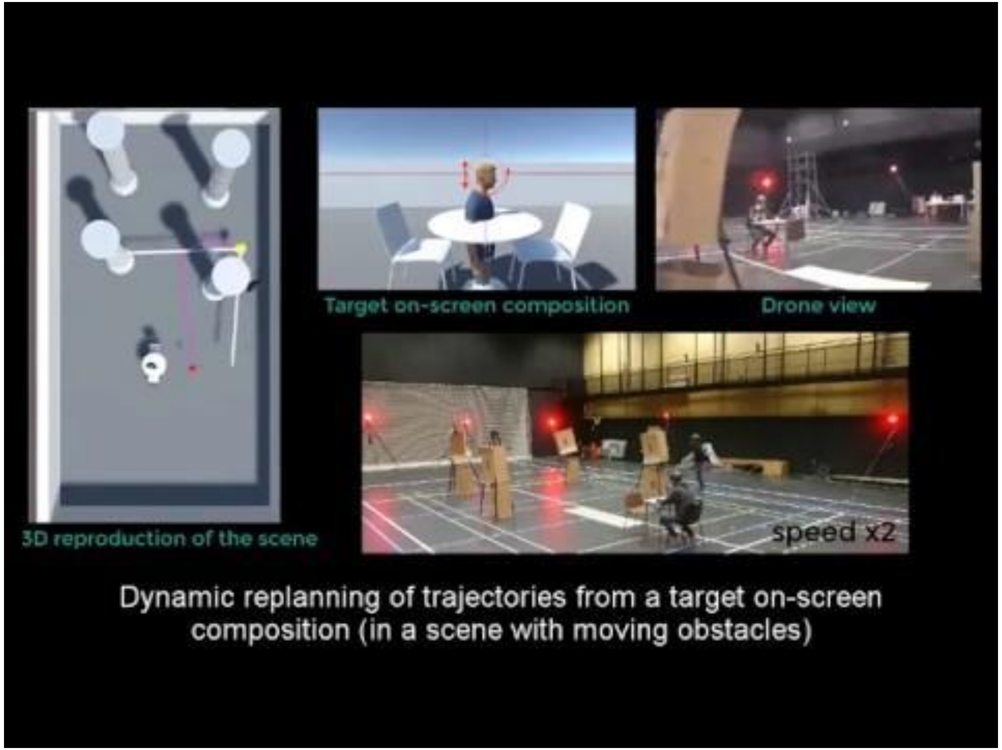
➢ Frustum culling in the roadmap







➢RESULTS


Target on-screen composition
Drone view
3D reproduction of the scene
speed x2
Dynamic replanning of trajectories from a target on-screen composition (in a scene with moving obstacles)

Discussion

Issues?

- Precise localisation (indoor / outdoor)
    - Using Ultra Wide Band technology?
    - Using robotics SLAM technology?

=> Yet, some outdoor scenario remain possible!

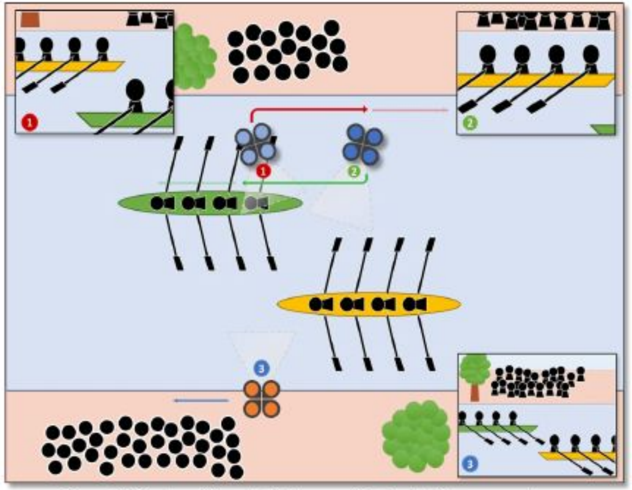


Figure 10: Overview of Scenario 2 - drones reacting to each other on approach

- Precise 3D representations for path planning and viewpoint quality
    - Use 3D reconstructed maps (photogrammetry)

But what’s next?

- Towards data-driven cinematography for drones (taking inspiration from real footage)
  - Extracting framing/motion features from sequences

Using DLIB tracker + OpenPose

Back to the Future

Back To The Future

A few words on what the future should be made of:

- Built-in tracking of cameras
  - Capture camera pose in real-time
  - Automated data extraction (shot time, actor, lighting)
  - Send data and meta-data to post-process
- No green screens!
  - automated contour extraction
- Volumetric performance capture of actors!
- Automated relighting of characters
  - Removing existing lighting
  - Adding new lighting

in real-time!

谢谢

你有问题吗？

(用英语讲）