Aalto University
School of Electrical
Engineering

Non-static object capture using multi-view stereo video

Konsta Hölttä

AS-0.3100 seminar presentation
konsta.holtta@aalto.fi

Apr 25, 2014

non-static object capture using multi-view stereo video

- Non-static / dynamic: moving or morphing over time
- $\Rightarrow$ video, maybe not just a simple object
- object: some target of interest in front of a camera
- capture: encode the object's visual properties digitally
  - geometry: 3d surface structure
  - texture: (diffuse) color
- multi-view stereo: recover 3D from photographs
- video: a sequence of frames

this presentation reviews the parts of the capture pipeline

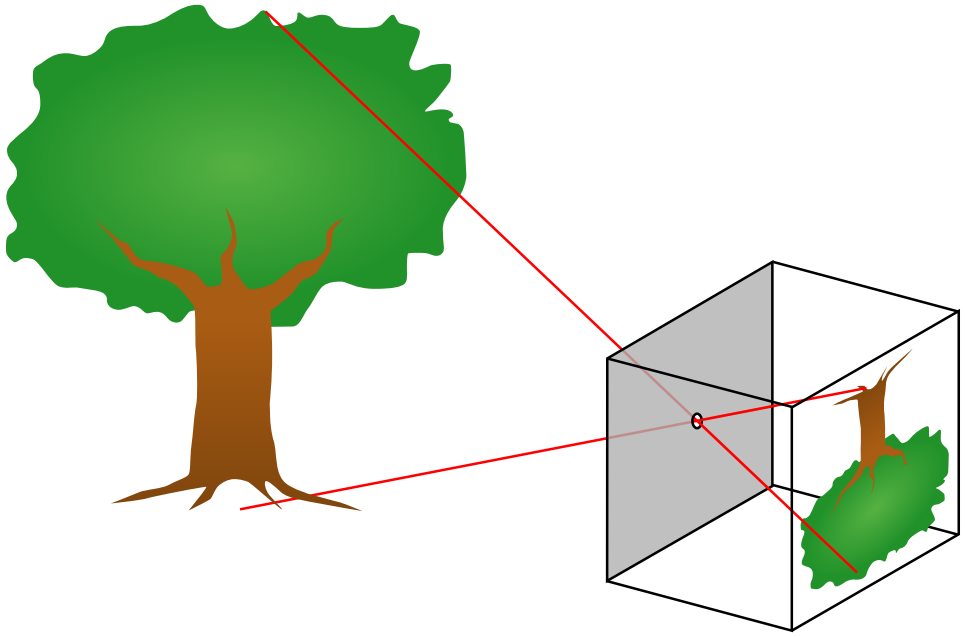
Contents

Motivation

L.A. Noire / joystiq.com

- applications in mapping, object replication, entertainment, cultural heritage, medical, crime investigation, ...
- most visible in modern movies and video games
- facial expressions; motion/surface capture
- relatively cheap hardware

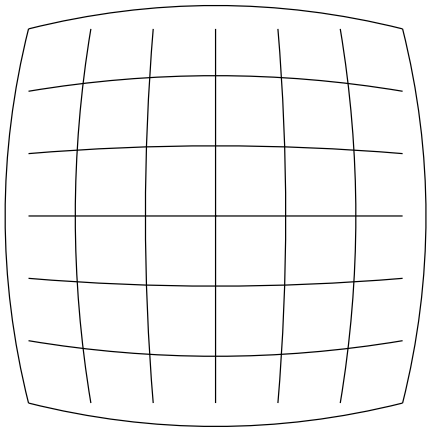
Cameras



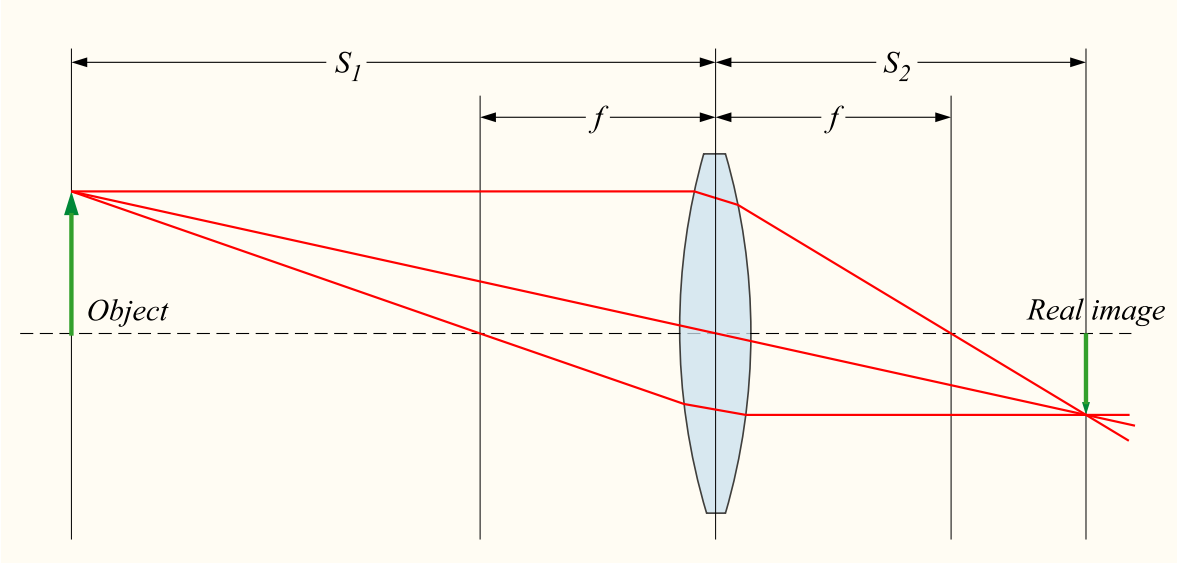
Pinhole camera projects an image perfectly (and rotated upside down)

$$\begin{pmatrix} u \\ v \end{pmatrix} = -\frac{f}{z} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Real cameras



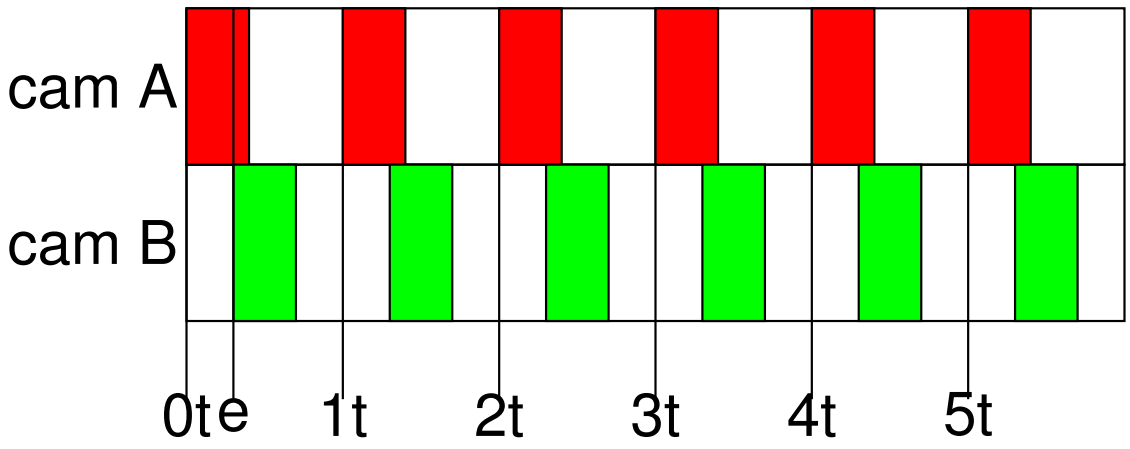
(a) barrel distortion



(b) Simple lens system (Wikipedia/DrBob)

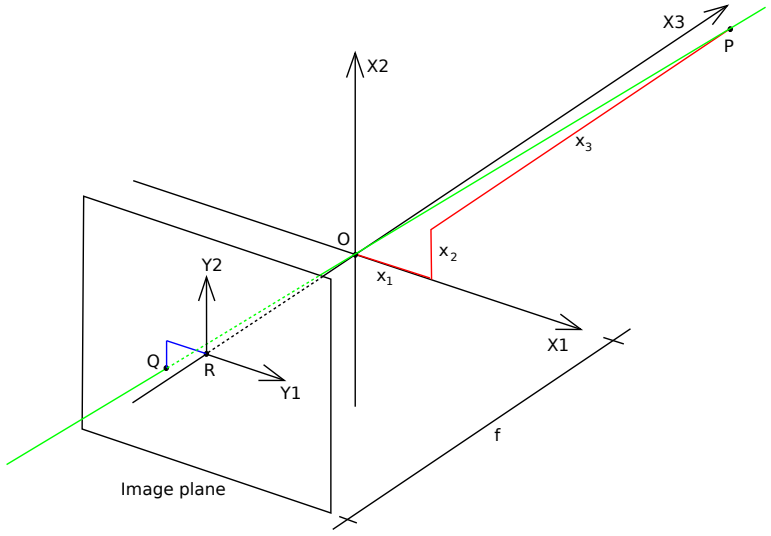
- Aperture size, depth of field, shutter speed, motion blur, diffraction, distortion, ...
- Capture hardware, image resolution, noise, frame rate, compression quality, ...
- Physical units, camera location, multi-camera calibration

Video concerns



- Lighting
- Motion blur
- Rolling shutter
- Sub-frame synchronization
- Other issues (23.97(6) FPS, mechanics, wiring, ..)

Calibration, camera parameters



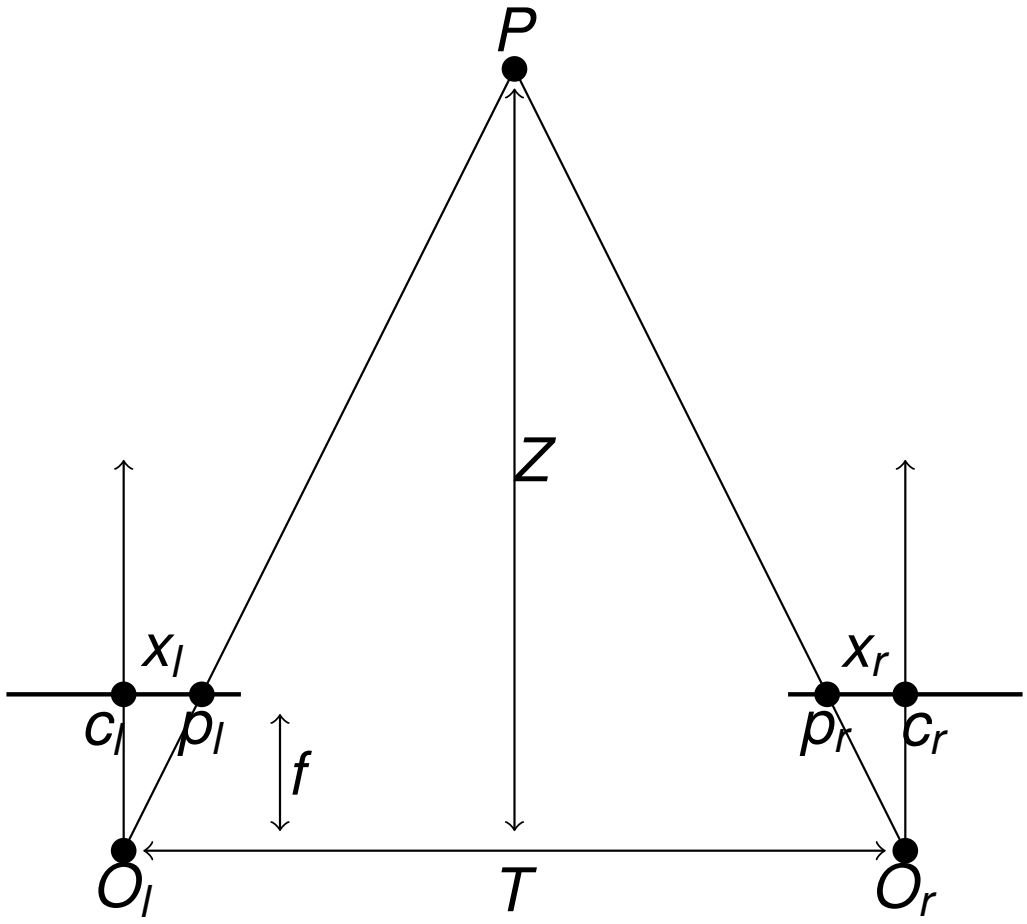
$$\begin{pmatrix} \alpha_x f & \gamma & u_0 \\ 0 & \alpha_y f & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Given known 3D coordinates and corresponding images, what is the projection matrix? (intrinsics, extrinsics)
- Done for single cameras and camera systems
- Projections closely related to computer graphics in general

(Multi view) stereo

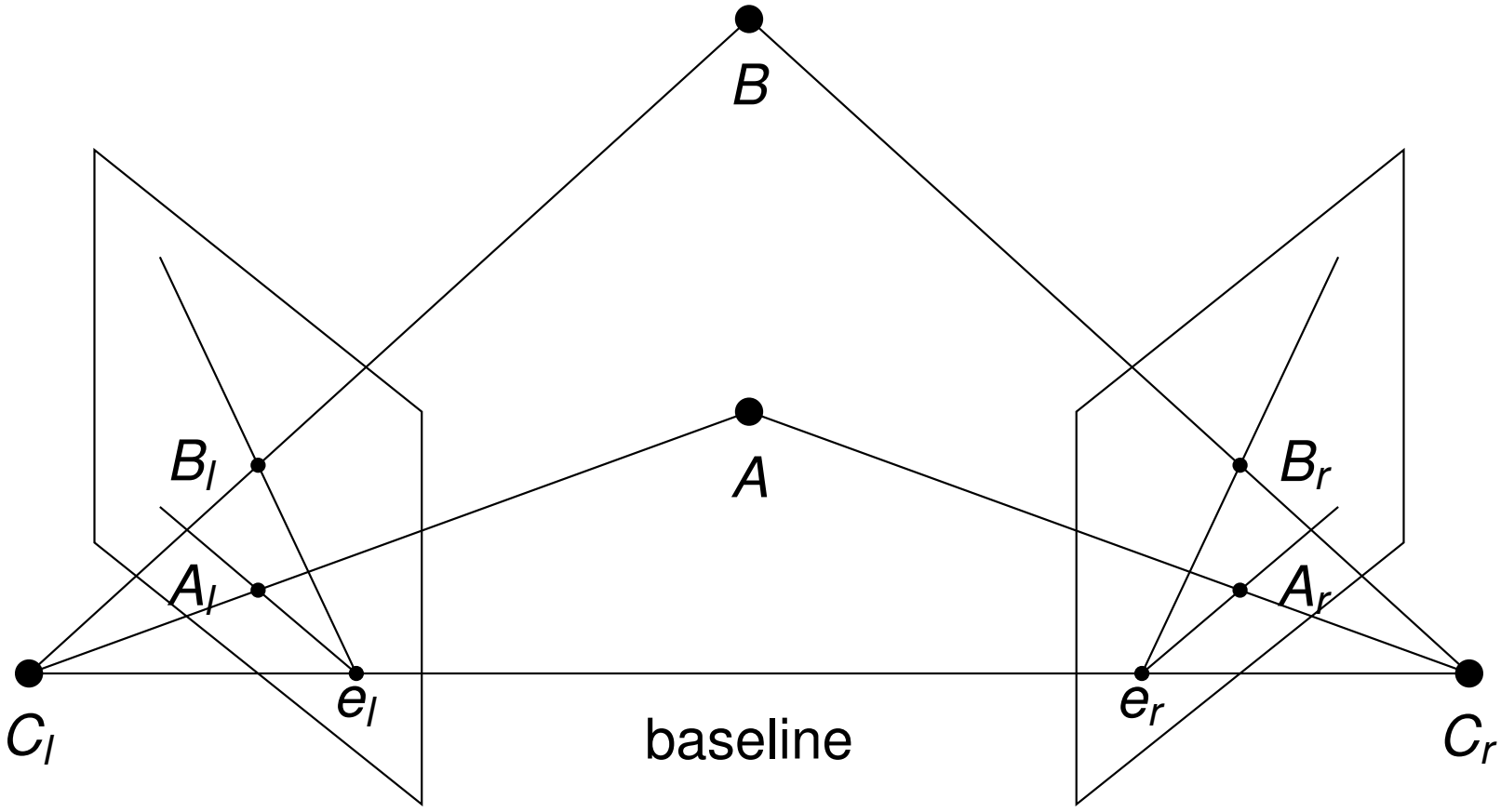
Stanford camera array.
Several cameras imaging the same scene at the same time

Disparity in stereo vision



Two identical cameras; depth inversely related to $x_r - x_l$, i.e. pixel disparity between corresponding points in images

Epipolar geometry



Corresponding points are found on lines. Rectification twists the images so that those lines become horizontal (or vertical).

Really multi-view

- Two camera case is still pretty dull
- Combine pairs, reconstruct individually, register in 3d
- Several on same baseline, match pairwise, fit least errors
- Use lots, arbitrarily positioned; magic algorithms
- Single camera traversing in a scene (SfM)

From static to dynamic

- Static can be scanned with a single camera only
- Moving targets need static frames from many angles
- Images must be consistent within a frame
- Stream of static frames reconstructed, or more sophisticated tracking

Dynamic methods

- Brute force reconstruction of every video frame
- Register together, build a mesh while going
- Morph initial model based on frame deformations
- Track local pre-selected features in 2D
- Track features in 3D, deform mesh based on keypoints
- And more and more post-processing, assumptions on the object properties

Texture and topology

- Reconstruction outputs local geometry properties
- Color per vertex, not textured, not meshed
- Reconstruct surface topology: fit data to surface, fill holes; use models
- Reproject camera views on ready mesh to recover texture maps
- Sometimes features tracked in 2D or 3D (still pre-recorded)
- Feature spaces and parameterizations in extreme cases

In practice

(EA sports, FIFA ’14; 18 DSLRs)

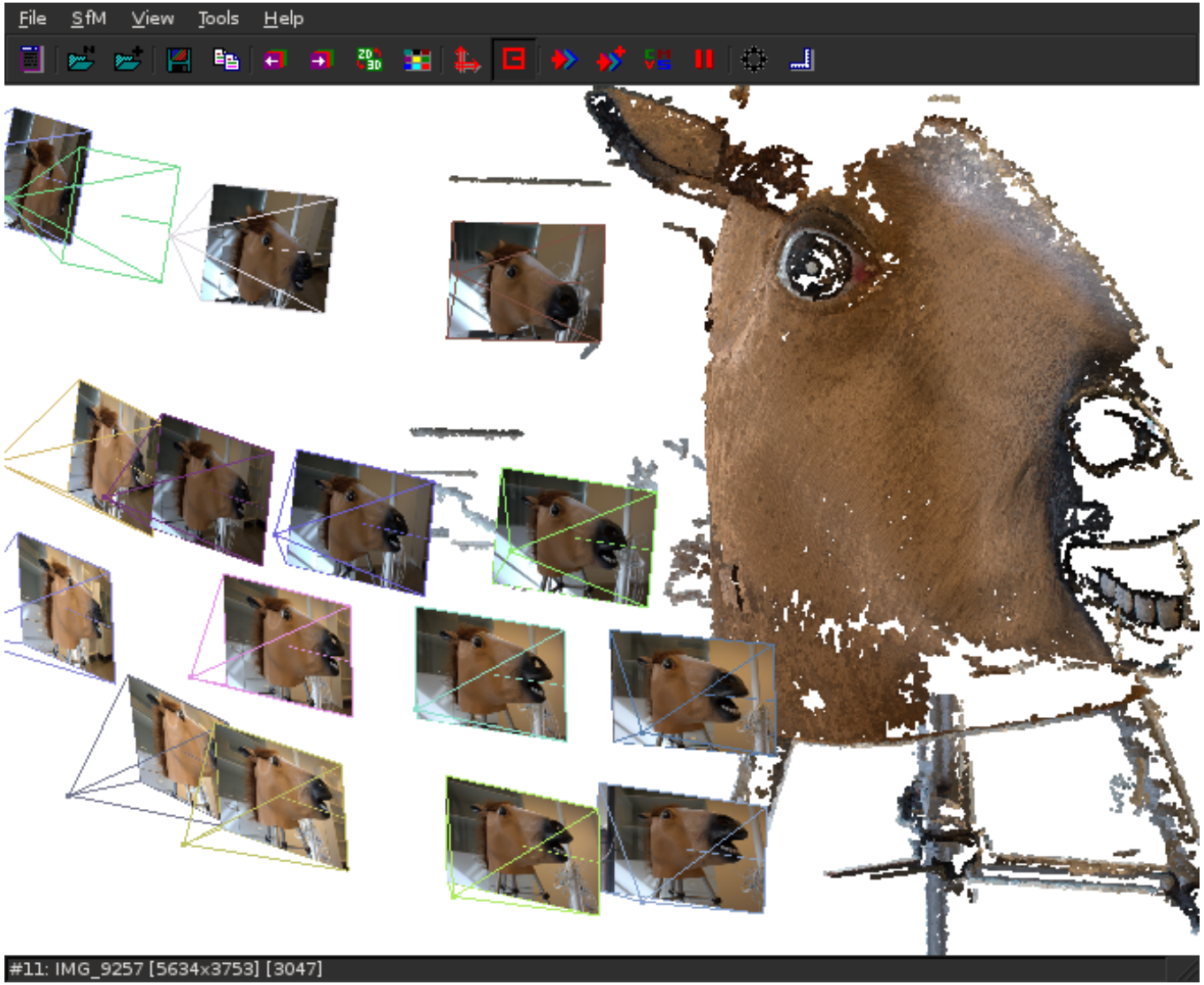
- Shutter delay and jitter, flash sync, remote control
- construction issues, calibration convenience, bad software, ...
- Manual work for artists, 3D noise removal, magic coefficients

Software

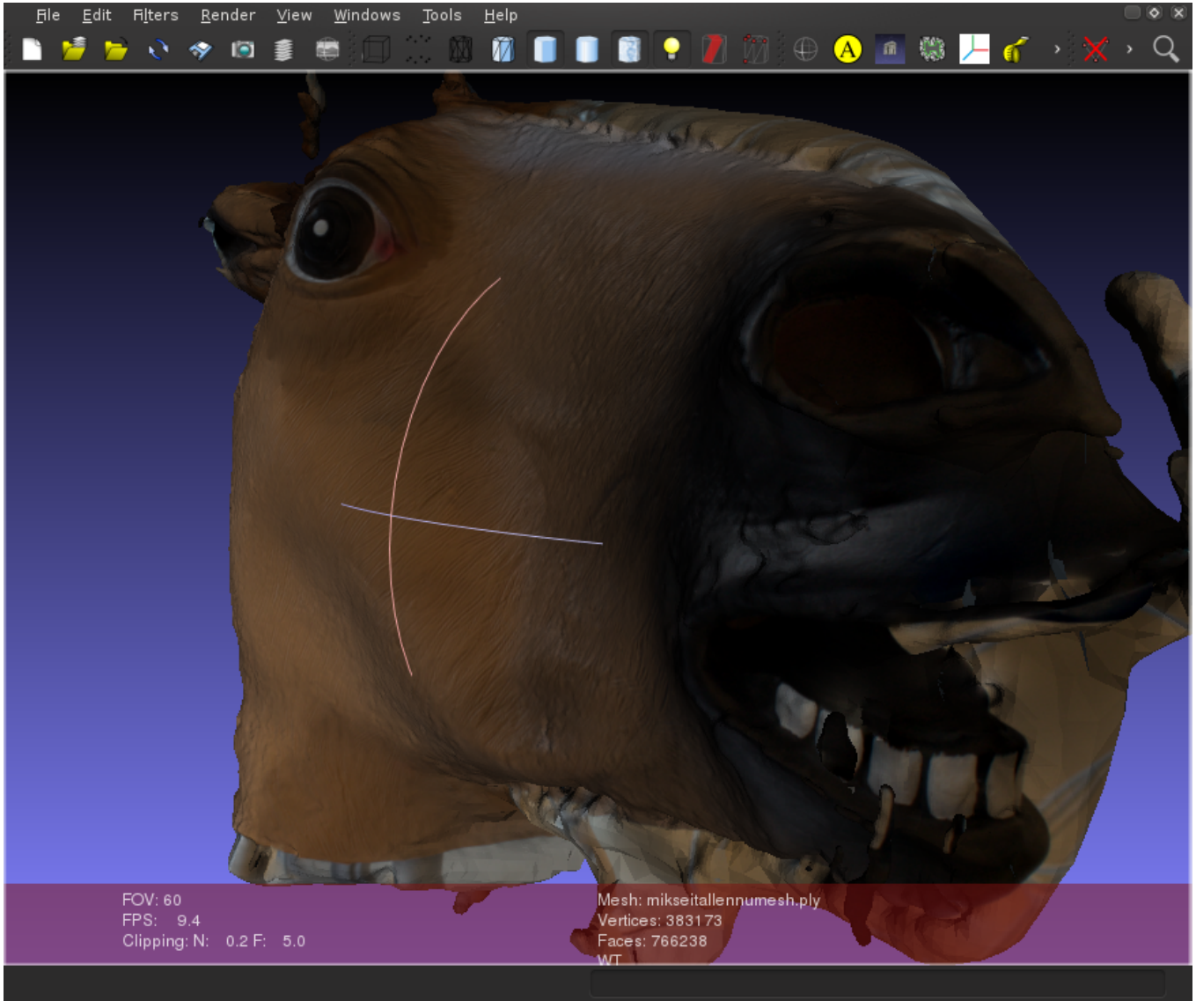
- Libs: OpenCV, Point Cloud Library, Computational Geometry Algorithms Library, ...
- Free: Camera calibration toolbox for Matlab (+OpenCV), Bundler, Sift, SiftGPU, PBA, PMVS/CMVS, VisualSFM, Meshlab, PoissonRecon, CmpMVS, Python photogrammetry toolbox, ...
- Commercial: Photosynth, 123D Catch, Agisoft PhotoScan, CaptiveMotion, DI3D, MotionScan, Faceshift, 3DF Zephyr Pro, Mova Contour Reality Capture, Pendulum Studio, Pix4DMapper, Acute3D, ...

4D programs very customized

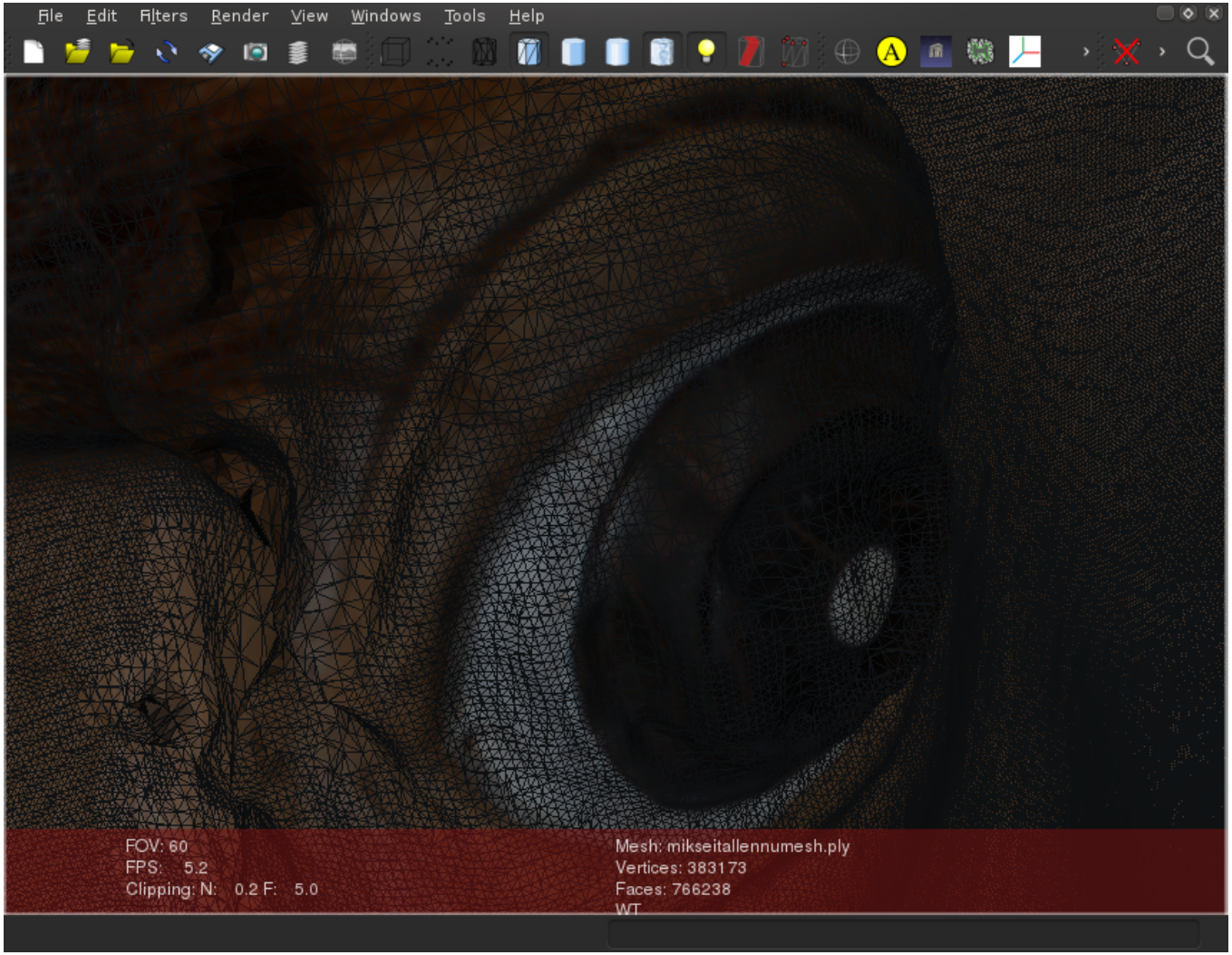
VisualSFM



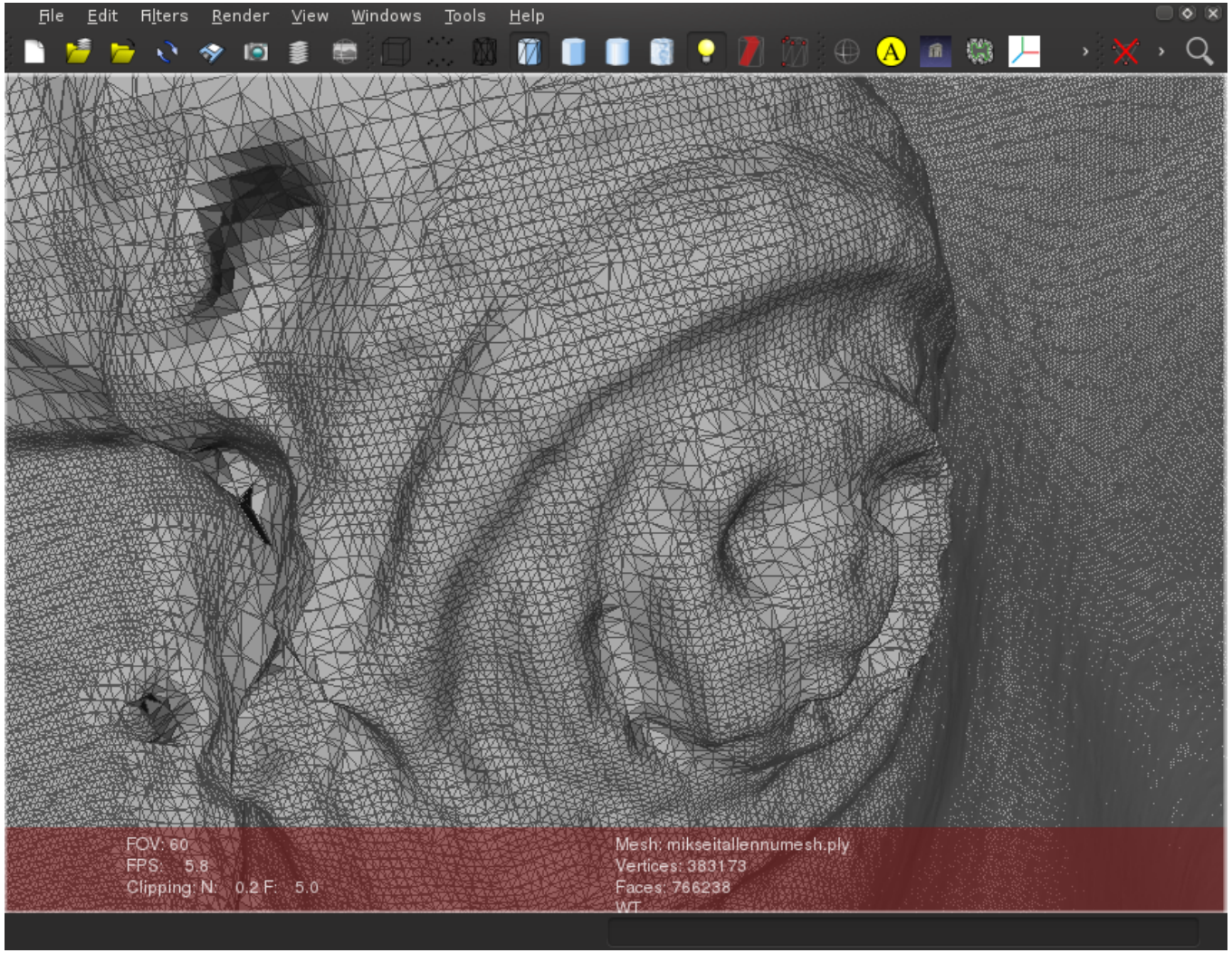
Meshlab


File Edit Filters Render View Windows Tools Help
FOV: 60
FPS: 9.4
Clipping: N: 0.2 F: 5.0
Mesh: mikseitallennumesh.ply
Vertices: 383173
Faces: 766238
WT

Meshlab


File Edit Filters Render View Windows Tools Help
FOV: 60
FPS: 5.2
Clipping: N: 0.2 F: 5.0
Mesh: mikseitallennumesh.ply
Vertices: 383173
Faces: 766238
WT

Meshlab


File Edit Filters Render View Windows Tools Help
FOV: 60
FPS: 5.8
Clipping: N: 0.2 F: 5.0
Mesh: mikseitallennumesh.ply
Vertices: 383173
Faces: 766238
WT

Conclusion

- 4D scan: multi-step pipeline from construction to shooting, reconstruction and post-processing
- Used method(s) largely application specific
- Lots of software, special applications coming, rising trend
- MVS just one scanning method

Many tools integrate some steps or add in their own