Mobile Augmented Reality

Wouter Pasman

July 5, 2004, Philips ApresLuvo Series

Delft University of Technology

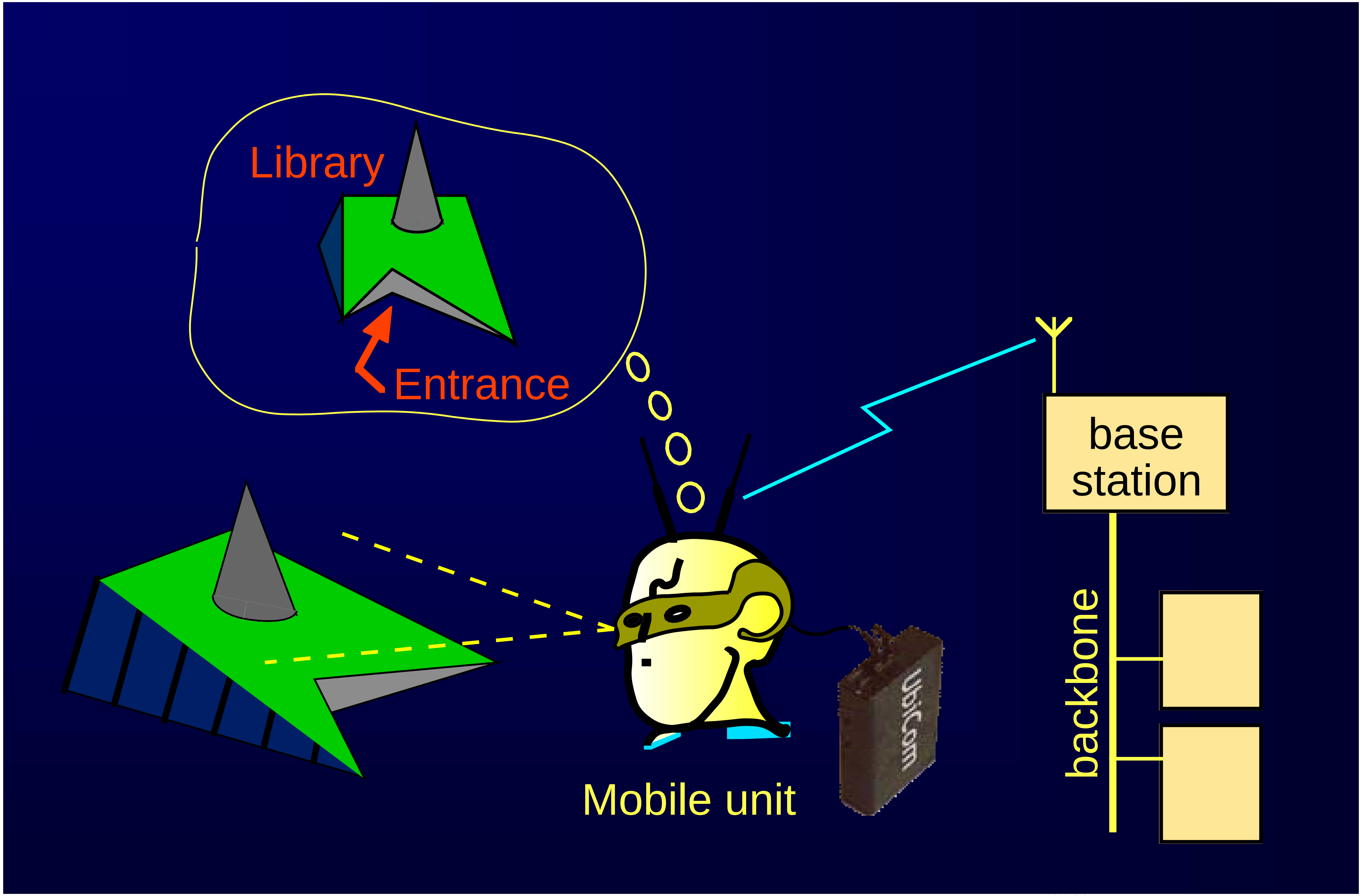
Ubiquitous Communications

Low Latency Mobile Augmented Reality

UbiCom

TU Delft

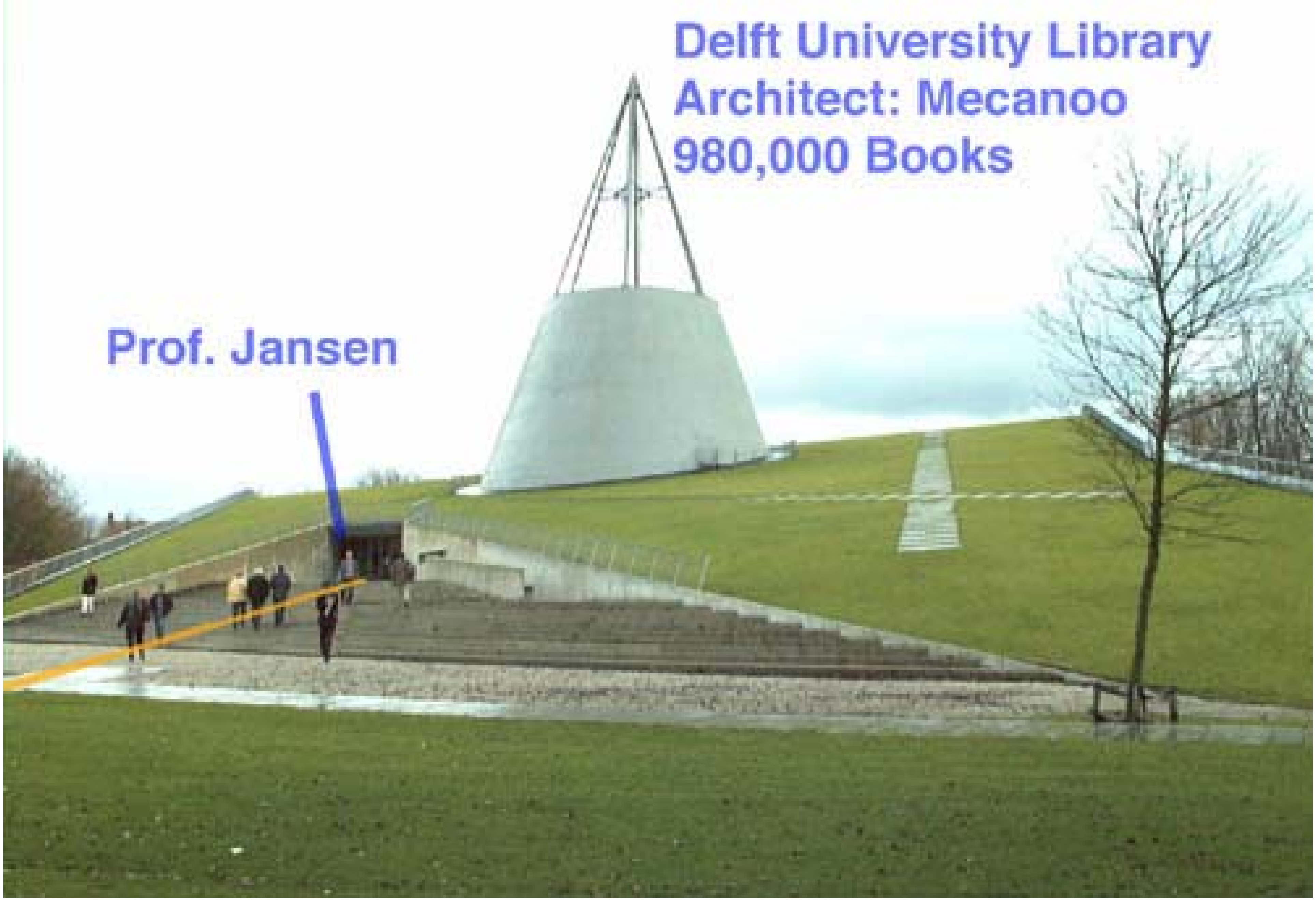
Delft University of Technology



UbiCom

TU Delft
Delft University of Technology

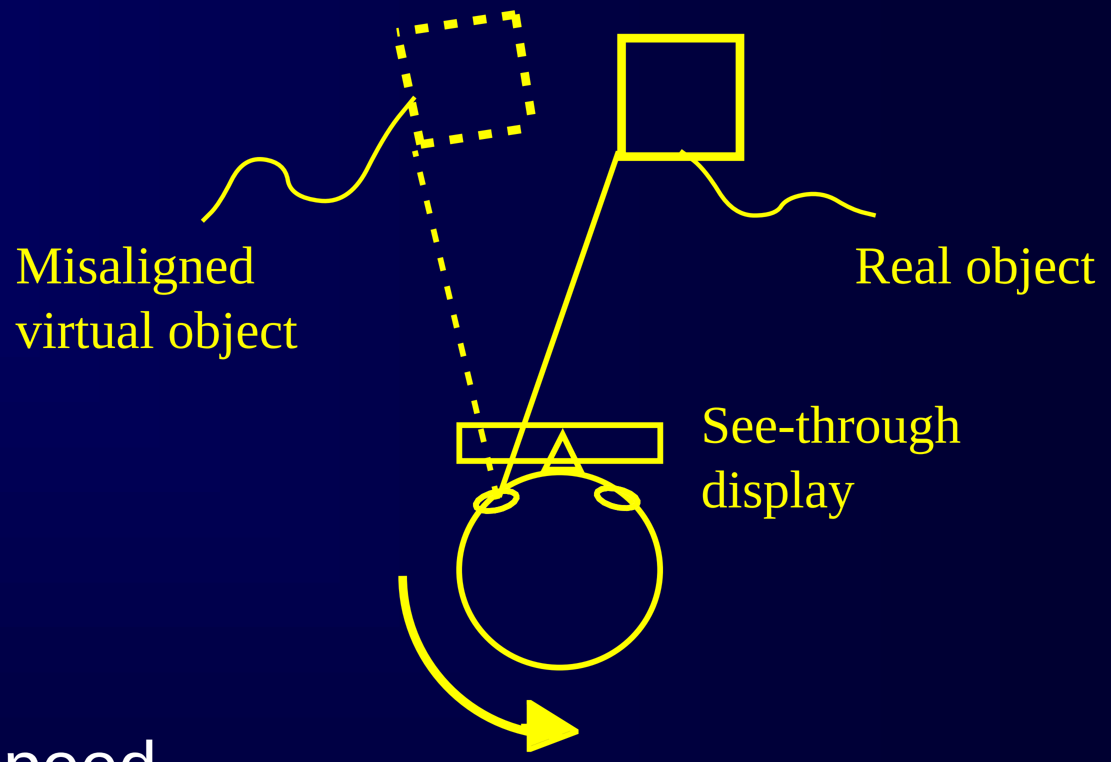
Applications




Augmented Reality
Virtual Reality

Maintenance, assistance

Latency in Optical AR



Alignment error=
Latency * Rotationspeed

For the applications targeted, 0.5° at 50°/s seems acceptable =>10ms.

UbiCom

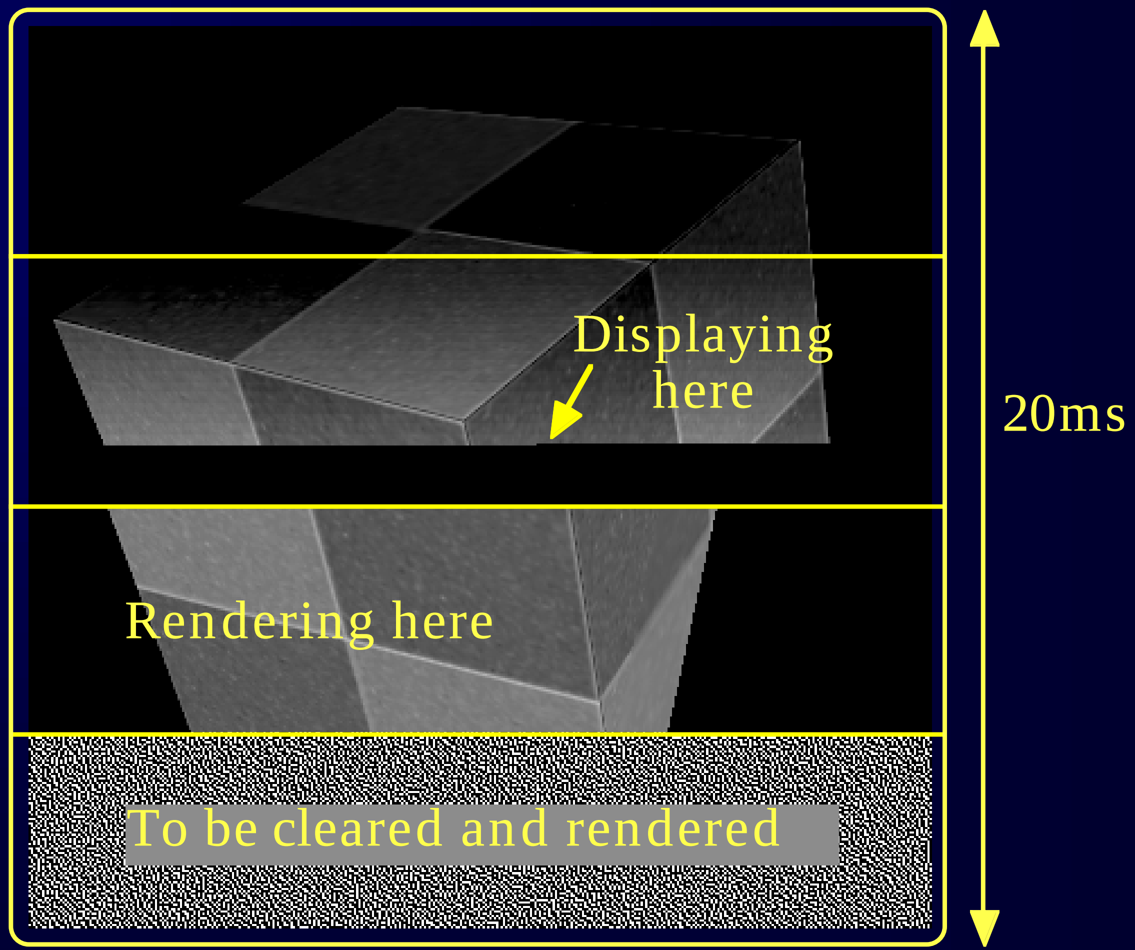
TUDelft
Delft University of Technology

Low-latency rendering

Std. Voodoo 3D
game card

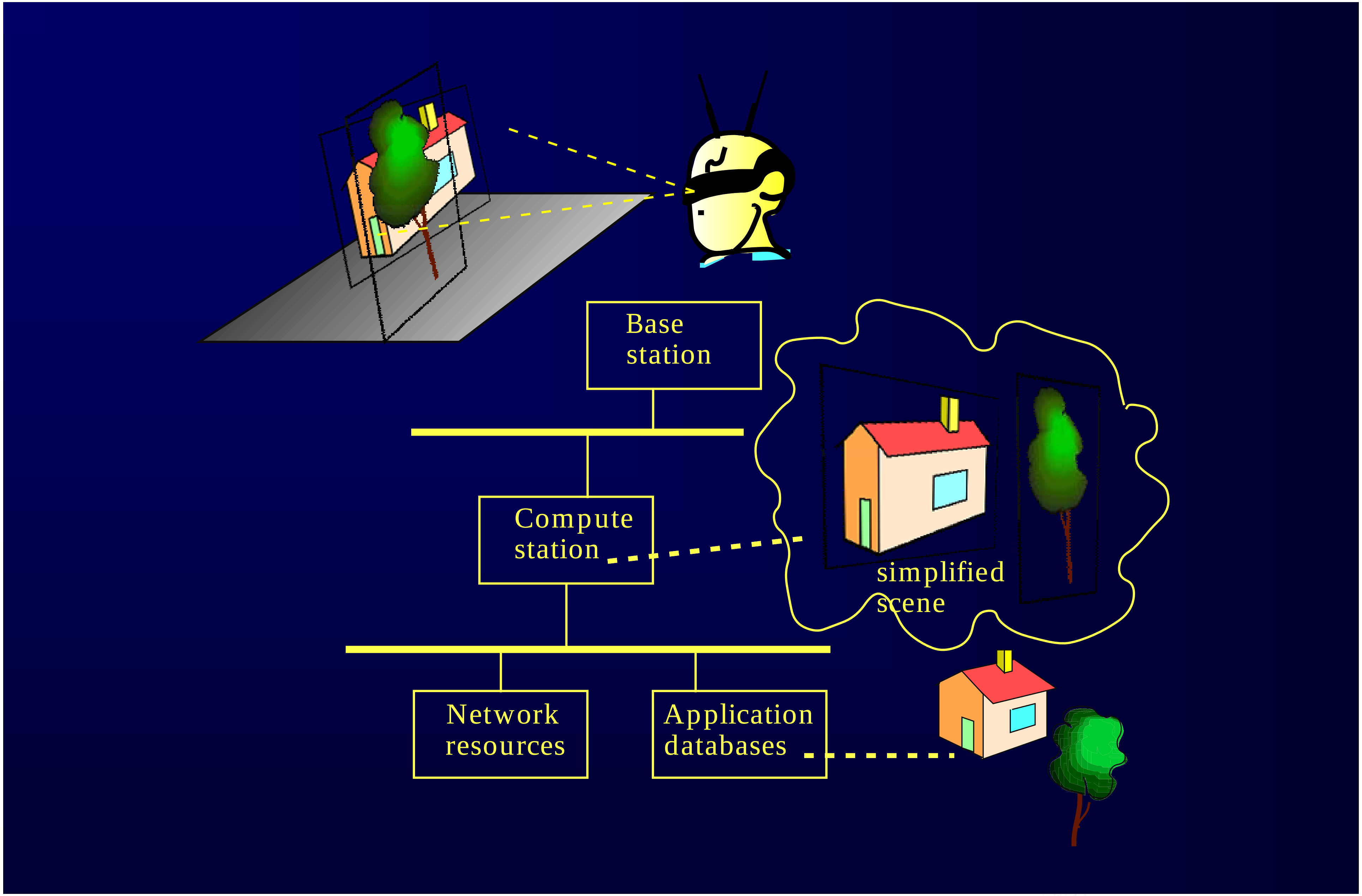
Render just ahead
of raster beam

4 partitions
gives latency
4-8 ms



UbiCom

TUDelft
Delft University of Technology

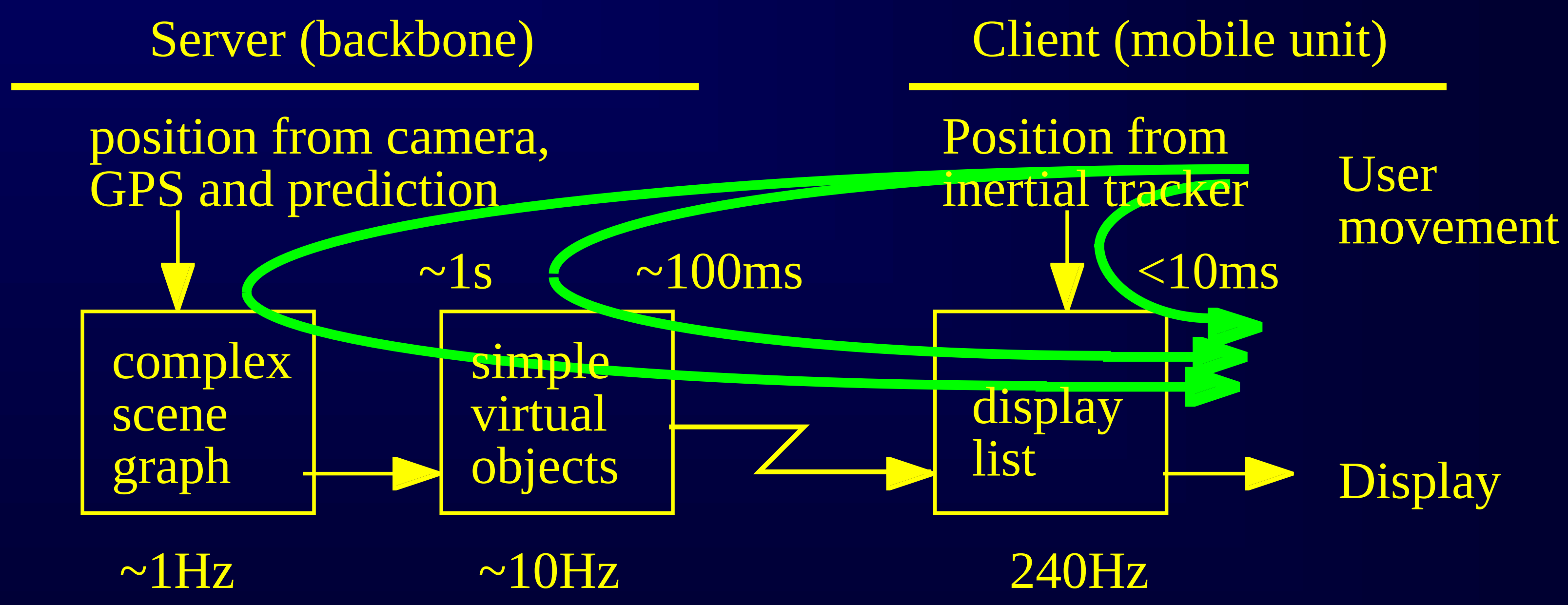


UbiCom

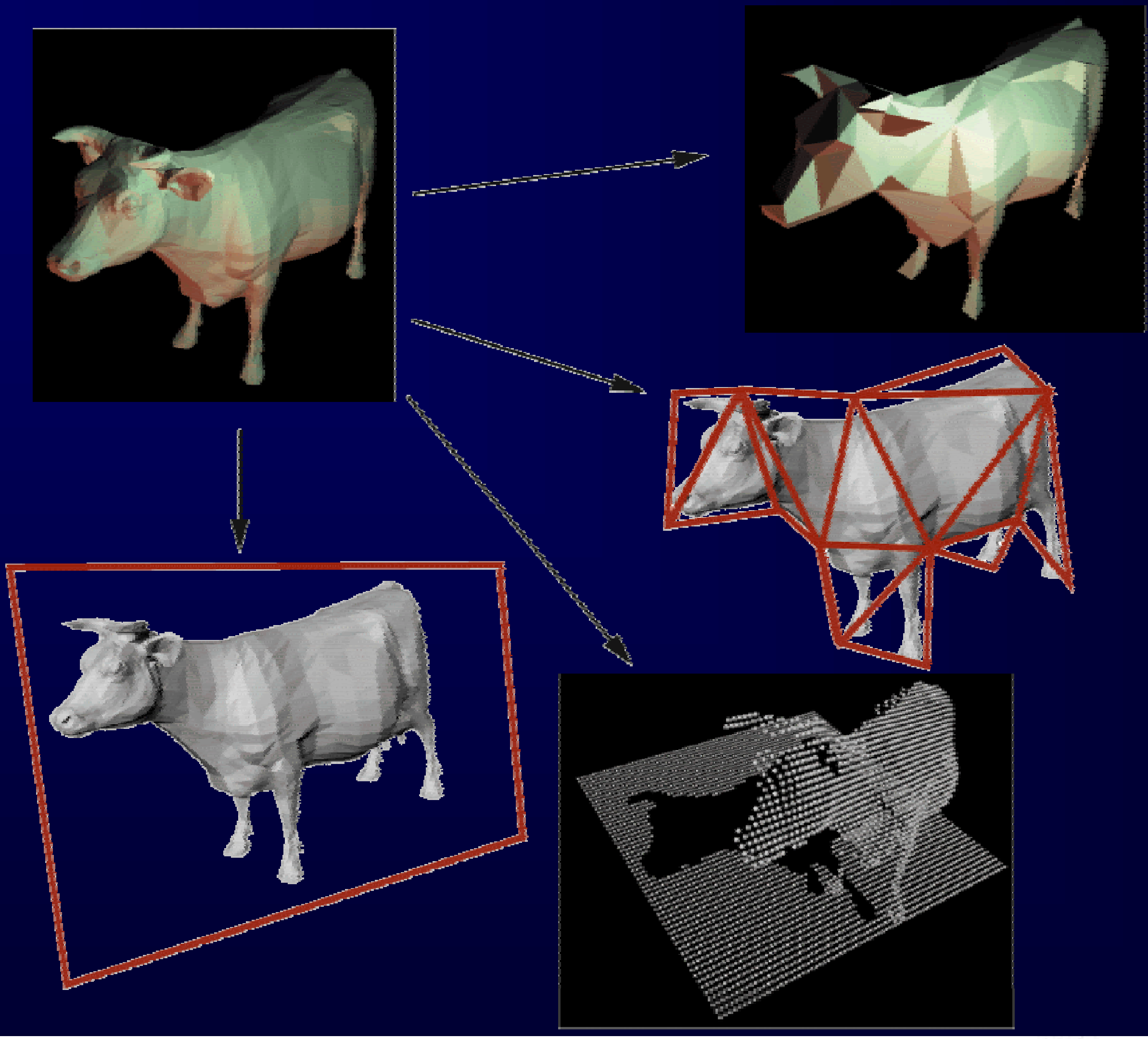
TU Delft
Delft University of Technology

Latency Layering

Limited resources on mobile, 250-400 polygons w. textures

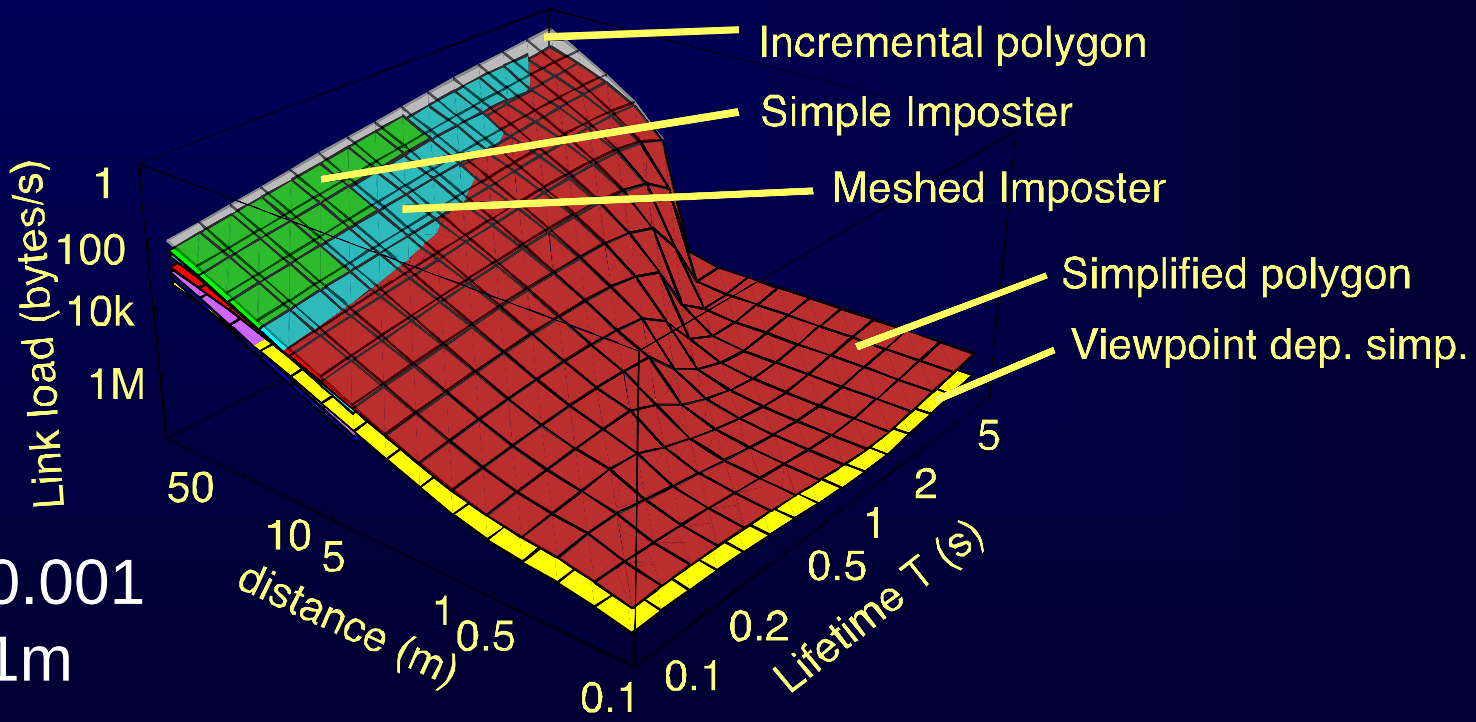


Dynamic Simplification

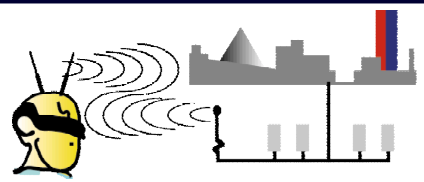


Mathematical model per object

- Estimate link and CPU load, memory usage, lifetime of objects, etc
- Est screenspace error and geometric distortions



D=0.001
R=1m



VIDEO: Statue on Campus

UbiCom

TU Delft

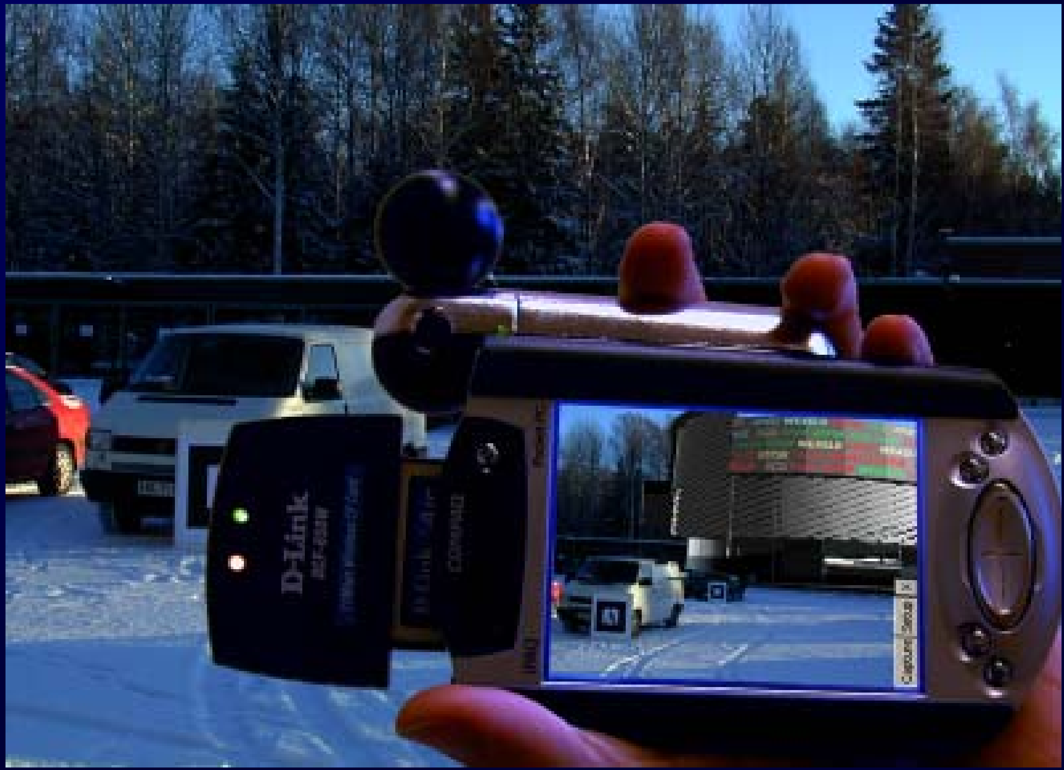
Delft University of Technology

NISHE

Augmented Reality with Large 3D Models on a PDA

Introduction

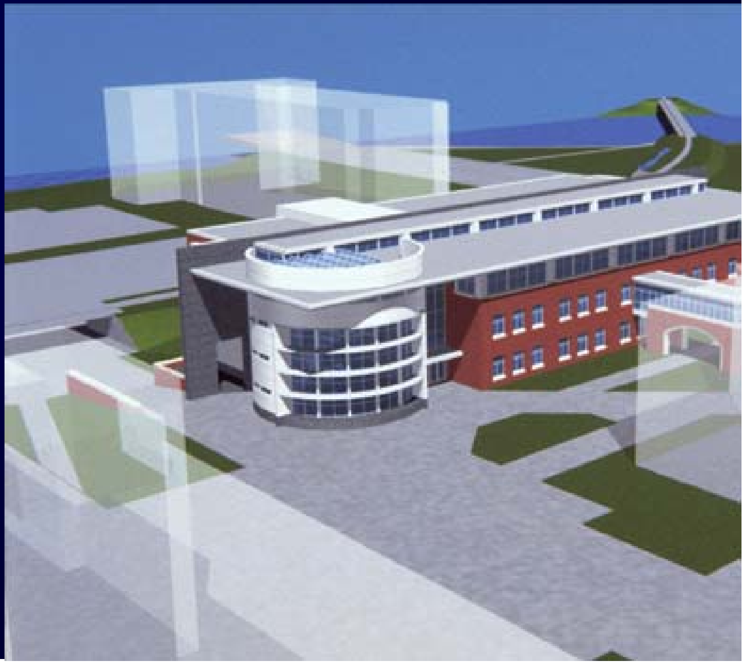
- AR with large models on PDA



Application area picked: supporting architects

VR is getting more popular for this.
But modeling of environment is cumbersome

--> often modeled quickly with large grey blocks



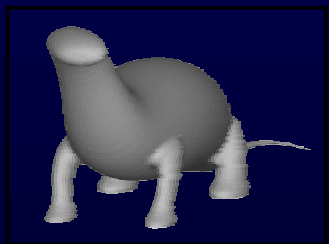
AR is making its way

-hand work: placing building at right location, proper lighting, occlusion, ...
- still picture

AR on PDA seems useful for such situations.

Architecture

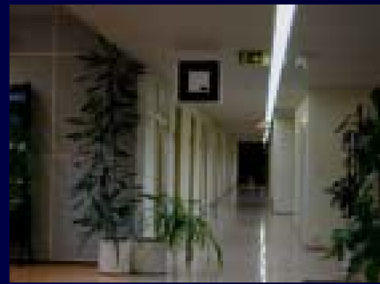
RLC decode, Track markers, Render virtual objects



Transparent bitmap of virtual objects

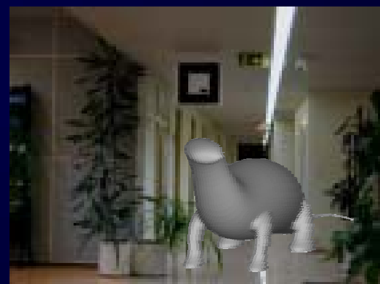
Server

PDA



Capture camera image

Show result



Decode virtual objects & mask, Overlay with camera image

Hardware:

PDA: iPaq H3800, Camera 640x240, display 240x320

206MHz StrongARM

Server: Dell Latitude, GeForce4 440 Go, 1.8GHz P4

Links tested: WLAN, USB, GPRS

Tracking

ARToolkit
Multimarking tracking: spanning large area with multiple markers
Markers 76cm wide for tracking up to 10m distance

ARToolkit adaptations:

- using low resolution 320x240 bitmap
- bitmap from link, not from camera
- Disable rendering of camera image

The Test Scene

Real scenes:

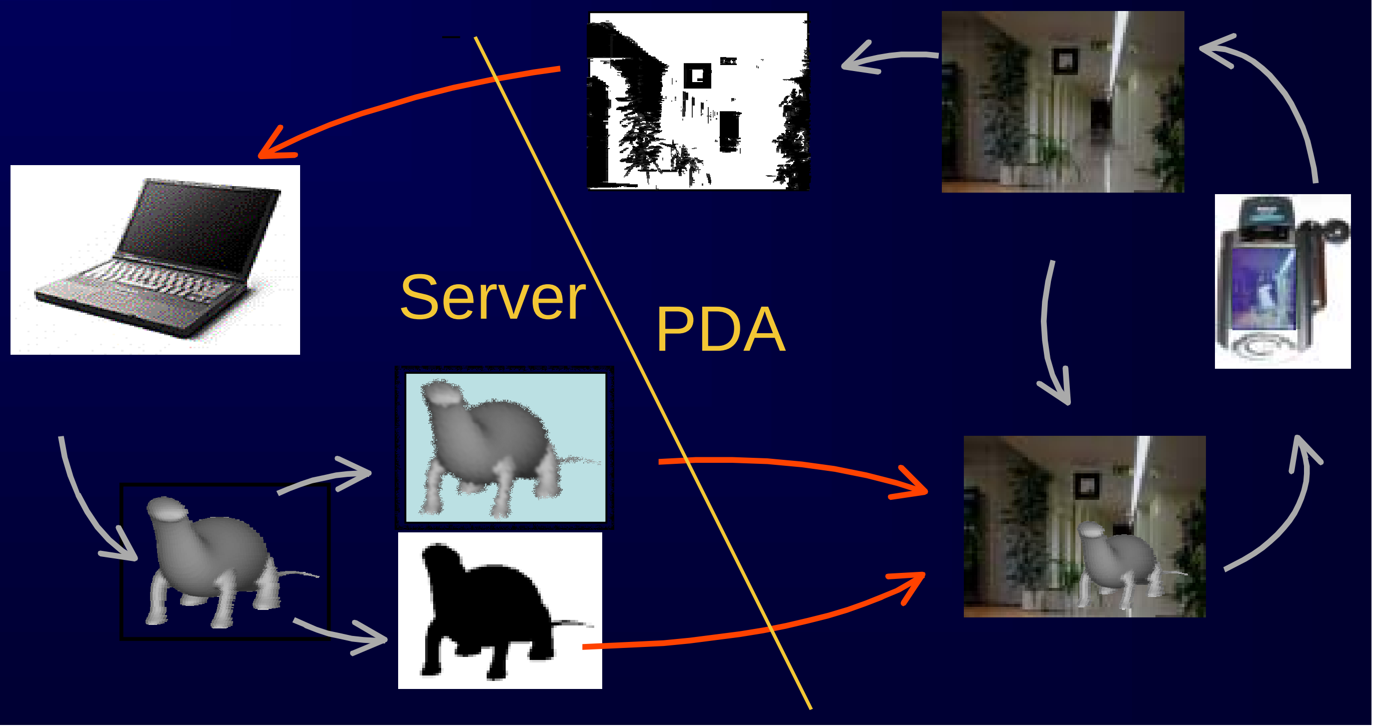
- outdoor parking place with snow, -20˚C, bright
  enhanced with few 76cm markers
- Lobby at entrance of the first floor
  enhanced with 40cm marker or
  with smaller markers as needed

Virtual scenes: VRML

- Simple scene (flower) not filling screen
- Itäkeskus building, 60k polygons w. texture
  60m wide, 15m high, more than screen filling

Compression Opportunities

1. Compressed B&W bitmap the camera image to the server
2. Video compress the overlay image to the PDA
3. Compressed Transparency mask to the PDA



- B&W bitmap the camera image to the server

- RGB to B/W: 24x compression
- RLE coding: using Elias Gamma code: 5x compression

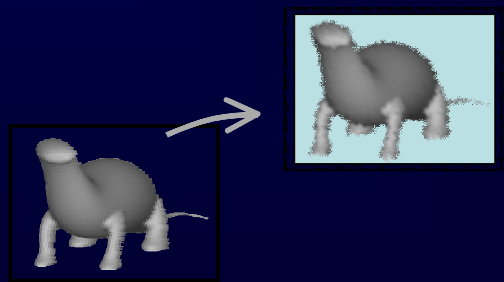
Cam image size:

Original 320x240	: 230 kbyte
B/W	: 9.6 kbyte
RLE coded	: 1.9 kbyte

2. Video encode the overlay image to the PDA

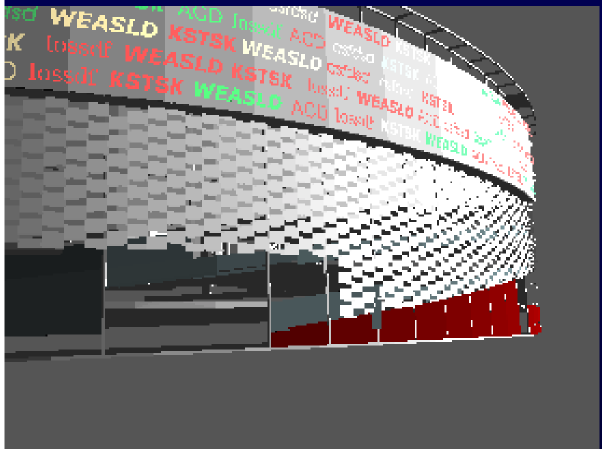
Using Motion Vector Quantization (MVQ)
Commercial coder, developed at our VTT group

- Very light decoding:
  - using motion vectors and lookup tables,
  - not using DCT
  - typically 50ms for full 320x240 image on PDA
- Large motion vectors up to 64 pixels,
  - suits shaky cam movements and low frame rates

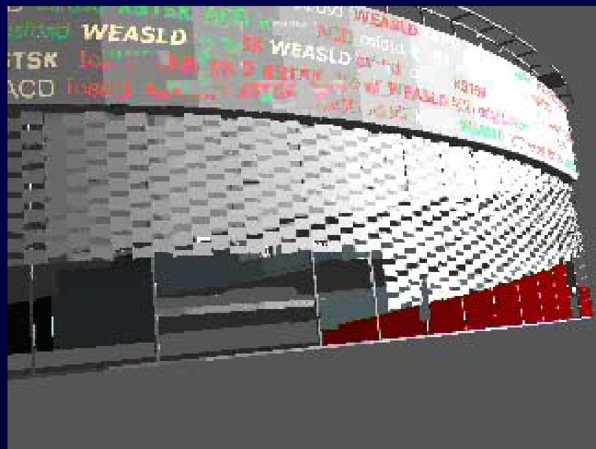


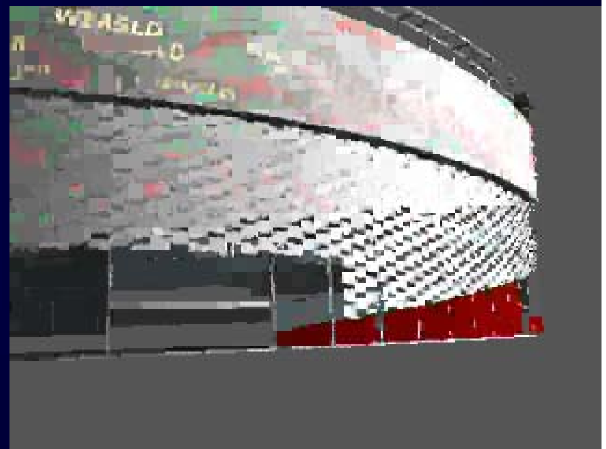
Optimizing MVQ Coding Modes
Optimization for Modem (4kb/frame) and Wavelan (30kb)

“Offline” = **Best** but 510ms/frame (10.8/15.3dB)
“Online” = **Fast** 160ms/frame but not so good (9.8/15.2dB)
Optimize for synthetic images with large smooth shaded areas
“Synthetic” = compromise, 200ms/frame (10.1/15.3dB)



Original

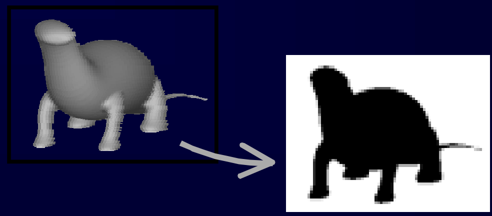

SNR 15dB


SNR 10dB

3. Compressed Transparency mask to the PDA

• RLE coding: using Elias Gamma code:
now 9x compression (less noise than natural imgs)

320x240 mask compresses about 1 kbyte.



Some Performance results

Without optimizations, “offline” MVQ compression, half-screen object, USB1 : 0.28 fps

With optimizations, worst case full screen object
- using USB1 and “online” : 0.9 fps
- using WLAN and “synthetic”: 1.25 fps
- using GSM and “synthetic” : 0.2 fps

Much more details in the paper.

Usability

- WLAN 1fps good for architecture. GSM is bit slow but convenient and always ready for demo
- Architects appreciate on-site experience of presence
- Need for markerless tracking
- ARToolkit has some tracking problems with certain marker orientations
- iPaq screen bit dim, especially when sunny
- Our system can be run even on mobile phone now.

Videos

- AR on PDA “digitalo”. (1:30)
- AR “indoors” (1:10)

Conclusions

- AR with video mixing was implemented on PDA/Mobile Phone.
- For mobile AR with optical mixing and for gaming latency is more critical. For such situations the UbiCom approach still seems the way to go.