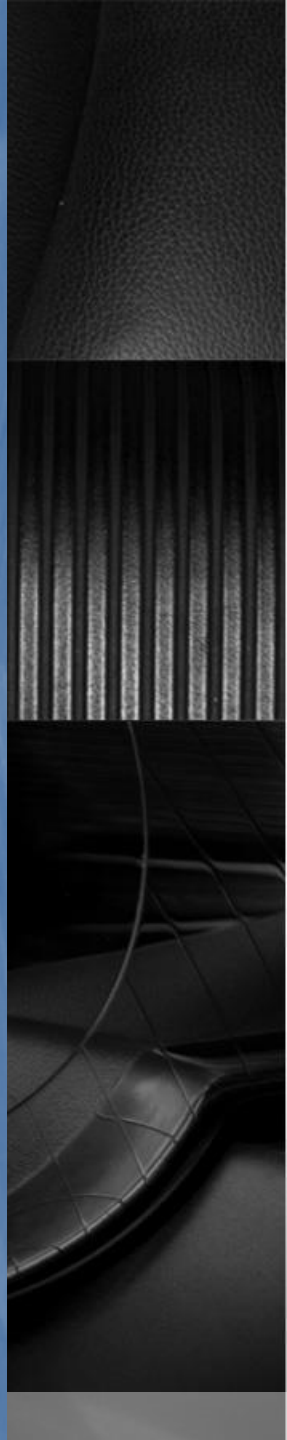
Web Based Graphics & Virtual Reality Systems

Web-based Graphics and Augmented Reality

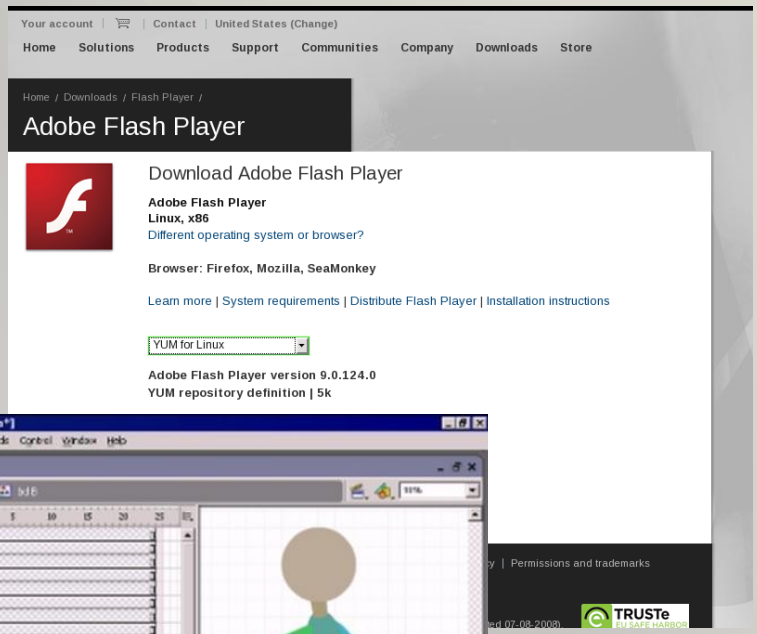


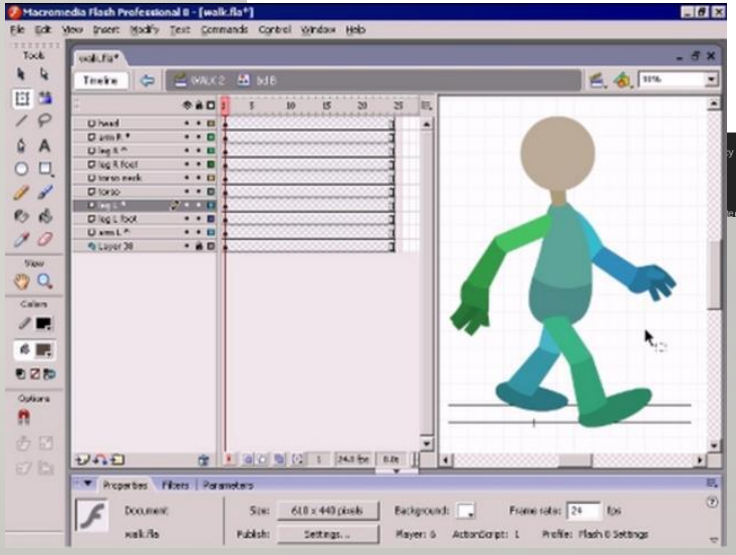
Graphics and the Web: The History

- World Wide Web starts with mainly textural content pages in the early 90s
- Only static images are supported in HTML
  - <img src="…">
- Late 90s, looped animated GIFs are first supported by Netscape Ver.2.0

Graphics and the Web: The History

- For a very long time, no further improvements of graphics supported in HTML
- Third party plugin becomes a solution to the inability of browser / HTML in 2000s
  - Java Applet
  - Flash and Sliverlight

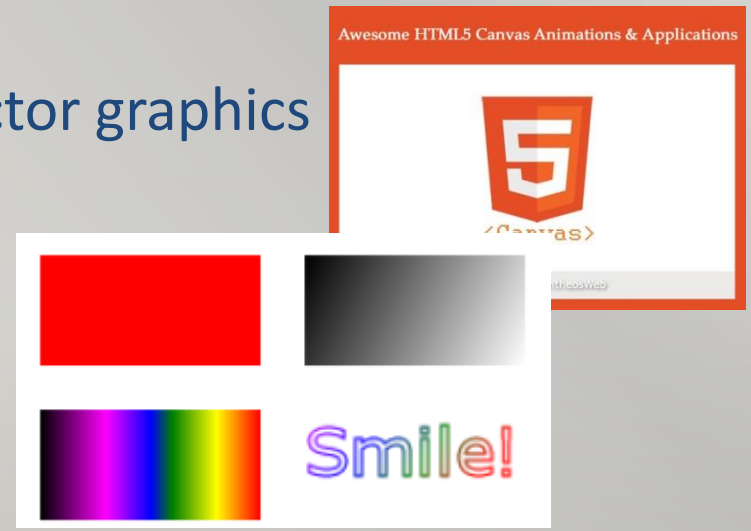




Graphics and the Web: The History

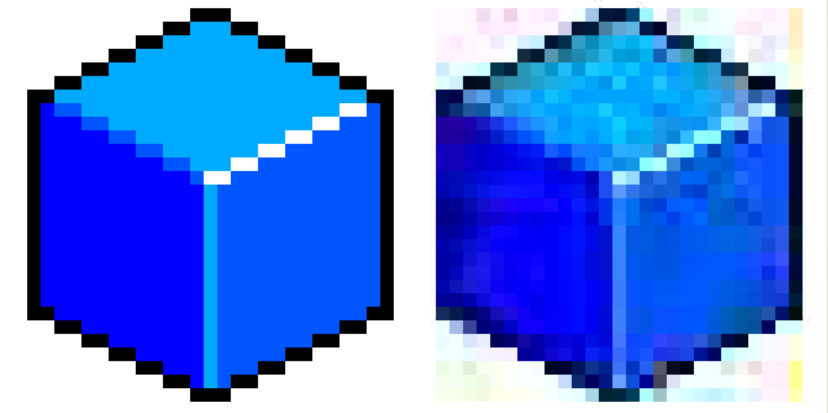
- Until HTML5, video playing is natively supported
  - By using <video> tag like :
    <video width="320" height="240" controls="controls">
    <source src="movie.mp4" type="video/mp4" />
    </video>

- Canvas is available for drawing vector graphics
  - <Canvas> …. </Canvas>
  - Details refers to tutorial note 1



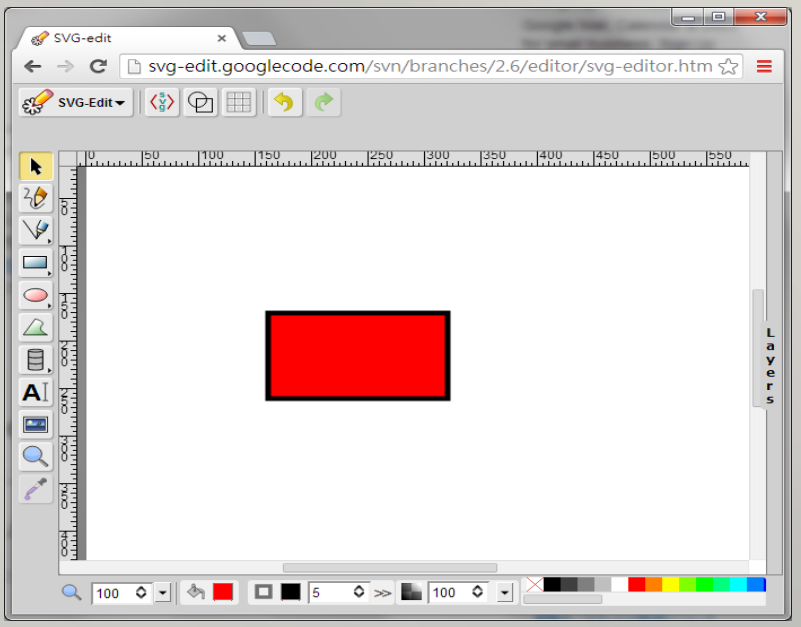
2D Graphics on Web

- Raster Images
  - Start from JPG, GIF
  - PNG, JPG2000
  - Lossless and lossy compression
- Vector Graphics
  - SVG (Scalable Vector Graphics
    - Defined in XML
  - HTML5 and Canvas



PNG: Lossless JPG: Lossy

http://commons.wikimedia.org/wiki/File:Cube_Gif_-_Cube_Jpeg.png



SVG-Edit (web-based SVG Editor)

http://svg-edit.googlecode.com/svn/branches/2.6/editor/svg-editor.html

Browser	JPEG	JPEG 2000	JPEG XR	GIF	PNG	APNG	MNG	TIFF [note 1]	SVG [note 2]	PDF [note 3]	2D Canvas	XBM	BMP
Amaya	Yes	?	No	Yes	Yes	No	No	Yes	Partial	No	No	?	?
AOL Explorer	Yes	No	No	Yes	Partial [note 4]	No	No	disable	No	No	No [note 5]	No [note 6]	?
Avant	Yes	No	No	Yes	Yes	No	No	disable	Partial	No	Yes	No [note 6]	?
Camino	Yes	No	No	Yes	Yes	No	No	No	Partial [note 7]	No	No	Yes	?
Chromium	Yes	?	?	Yes	Yes	No [note 8]	?	?	Partial	No	Yes	?	Yes
Dillo	Yes	No	No	Yes	Yes	No	No	No	No	No	No	No	?
ELinks	No	No	No	No	No	No	No	No	No	No	No	No	?
Web	Yes	No	No	Yes	Yes	Dropped	No	No	Partial	No [note 9]	Yes [note 10]	Yes	?
Flock	Yes	No	No	Yes	Yes	Dropped	No	No	Partial [note 7]	No	Yes	Yes	?
Galeon	Yes	No	No	Yes	Yes	Yes	No	No	No	No	No	Yes	?
Google Chrome	Yes	No	No[47]	Yes	Yes	No	No	No	Partial	Yes	Yes	Yes	Yes
iCab	Yes	?	No	Yes	Yes	No	No	No	No	Yes	No	?	?
Internet Explorer	Yes	No	Yes	Yes	Yes	No	No	Yes	Partial	No	Yes ,[48][note 5]	Dropped [note 6]	Yes
Internet Explorer for Mac	Yes	No	No	Yes	Yes	No	No	No	No	No	No	?	?
K-Meleon	Yes	No	No	Yes	Yes	No	No	No	No	No	No	?	?
Konqueror	Yes	Yes [note 11]	No	Yes	Yes	No	Yes	Yes	Partial [note 12]	Yes [note 13]	Yes	?	Yes
Links	Yes	?	No	Yes	Yes	No	No	Yes	No	No	No	Yes	?
Lynx	No	No	No	No	No	No	No	No	No	No	No	No	No
Maxthon	Yes	No	No	Yes	Yes	No	No	disable	Partial	No	Yes	No [note 6]	Yes
Midori	Yes	Yes [note 14]	No	Yes	Yes	No	No	Yes	Partial [note 15]	No	Yes	Yes	?
Mosaic	Yes	No	No	Yes	Yes	No	No	No	No	No	No	Yes	?
Mozilla	Yes	No	No	Yes	Yes	No	Dropped [note 16]	No	No	No	No	Yes	?
Mozilla Firefox	Yes	No[49]	No[50]	Yes	Yes	Yes [note 17]	No[51]	No[52]	Partial [note 7]	Yes	Yes	Dropped [note 18]	Yes
Netscape	Yes	No	No	Yes	Yes	No	Dropped [note 19]	No	No	No	No	Yes	?
Netscape Browser	Yes	No	No	Yes	Depends [38]	No	No	No	No	No	No	Yes	?
Netscape Navigator	Yes	No	No	Partial	32-bit version only	No	No	No	No	No	No	Yes	?
Netscape Navigator 9	Yes	No	No	Yes	Yes	No	No	No	Partial	No	Yes	Yes	?
NetSurf	Yes	No	No	Yes	Yes	No	Yes	No	Partial	No	No	No	?
OmniWeb	Yes	Yes	No	Yes	Yes	No	No	Yes	Partial	No	No	Yes	?
Opera	Yes	No	No	Yes	Yes	Dropped	No	No	Partial [note 20]	No	Yes	Yes	Yes
QupZilla	Yes	Yes	No	Yes	Yes	No	No	Yes	Partial [note 21]	No	Yes	Yes	Yes
Safari	Yes	Yes [note 14]	No	Yes	Yes	No	No	Yes	Partial [note 15]	Yes (Mac only)	Yes	Yes	Yes

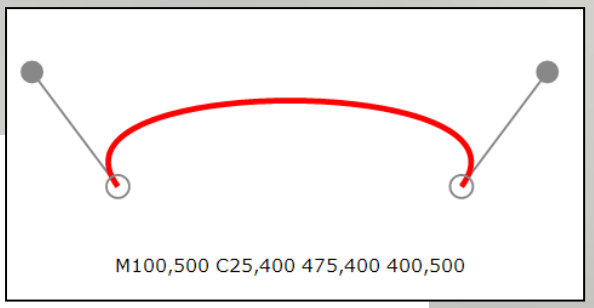
http://en.wikipedia.org/wiki/Comparison_of_web_browsers#Image_format_support

SVG (Scalable Vector Graphics)

- Web standard of vector graphics
- Defined in XML format
- Shape, dimension, color and etc.
- Example
  - <rect > for filled rectangule
  - <path > for line and curves

    M is moveto , C is “curveto” / control points

```
<?xml version="1.0"?>
<svg width="400" height="110">
  <rect width="300" height="100"
style="fill:rgb(0,0,255);stroke-
width:3;stroke:rgb(0,0,0)" />
</svg>
```



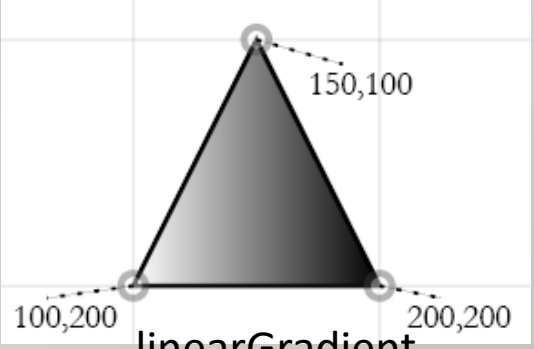

```
…
  <path class="SamplePath"
d="M100,500 C25,400 475,400 400,500" />
…
```

http://www.w3.org/TR/SVG11/paths.html

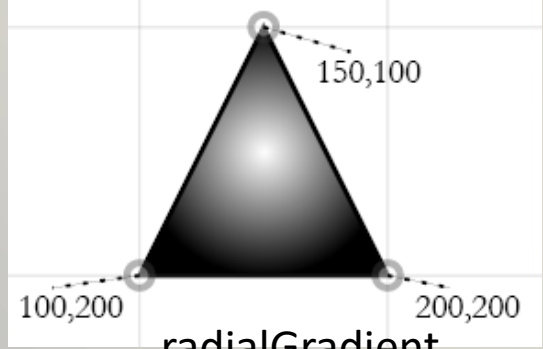
SVG (Scalable Vector Graphics)

Other features

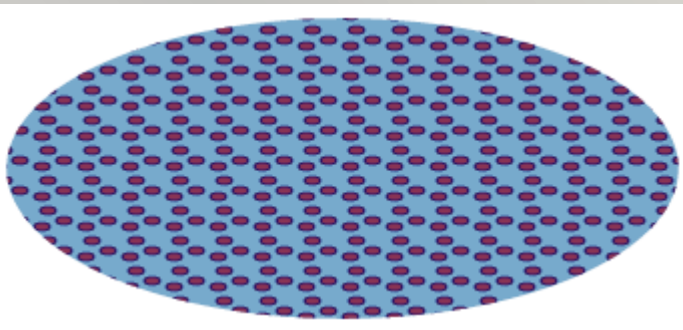
- gradients
  - <linearGradient />
    <radialGradient />
- Embed images
- Blur and filtering
  - <filter />
- Pattern
  - <pattern />
- No native support to interactions and animation
- Interaction still can be done via external scripting



linearGradient



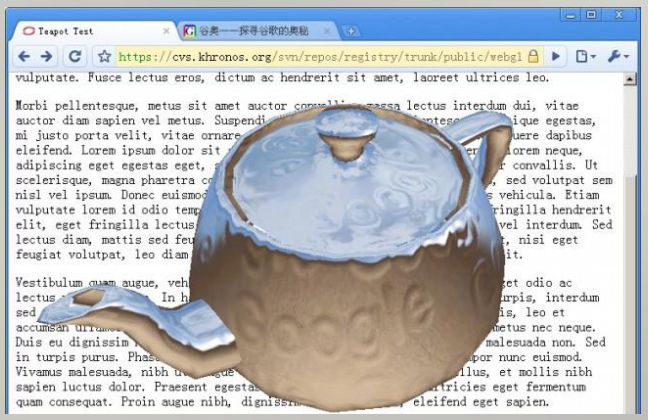
radialGradient



http://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

3D Graphics on Web

- In early 90s, none of web browser support 3D rendering
- In late 90s, Java3D and VRML are released
  - Scene graph application
- Other commercial products
  - Macromedia/Adobe Shockwave 3D
  - Unity
- In late 2000, WebGL emerged
  - A porting of OpenGL|ES standard to web browsers



Web-based 3D Graphics Technologies

VRML

O3D

Java 3D

X3D

JOGL

Solutions for Web-based 3D graphics

Burster

And more. . .

Unity

WebGL

Adobe Shockwave

VRML and X3D

- Organized and consolidated the standard by Web3D Consortium
- International Organization of Standardization (ISO) have made X3D an approved standard for real-world use, both on and off the Web
- X3D : XML encoded version of VRML, so it supports
  - XML Compressed Binary encoding
  - XML Encryption
  - XML Digital Signature (for authentication)

Features of X3D

- 3D graphics
  - Polygonal geometry, parametric geometry, hierarchical transformations, lighting, materials and multi-pass/multi-stage texture mapping
- 2D graphics
  - Text, 2D vector and planar shapes displayed within the 3D transformation hierarchy
- Animation
  - Timers and interpolators to drive continuous animations; humanoid animation and morphing
- Spatialized audio and video
  - Audiovisual sources mapped onto geometry in the scene
- User interaction
  - Mouse-based picking and dragging; keyboard input
- Navigation
  - Cameras; user movement within the 3D scene; collision, proximity and visibility detection

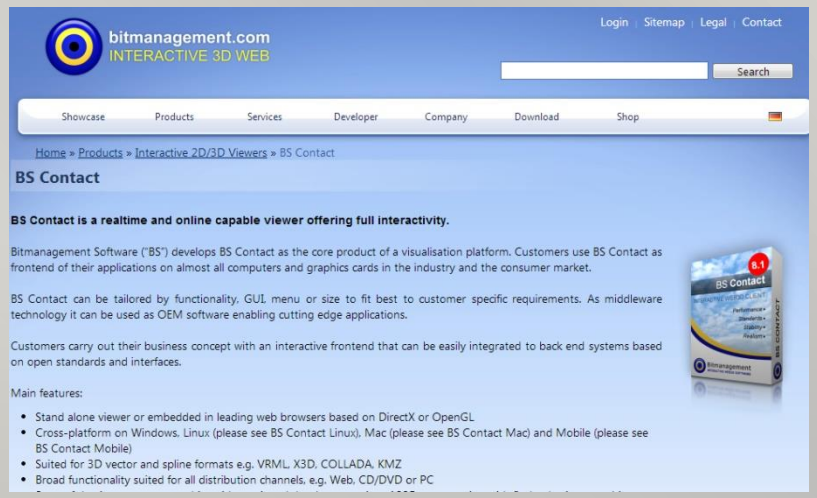
http://www.web3d.org/files/specifications/19775-1/V3.3/index.html

Features of X3D (cont')

- User-defined objects
  - Ability to extend built-in browser functionality by creating user-defined data types
- Scripting
  - Ability to dynamically change the scene via programming and scripting languages
- Networking
  - Ability to compose a single X3D scene out of assets located on a network; hyperlinking of objects to other scenes or assets located on the World Wide Web
- Physical simulation
  - Humanoid animation; geospatial datasets; integration with Distributed Interactive Simulation (DIS) protocols
- Geospatial positioning
  - Ability to accurately position X3D scene objects geospatially.
- CAD geometry
  - ability to represent CAD models mapped from CAD systems.
- Layering
  - Ability to organize X3D scenes into rendering groups so that objects in each layer can overlay objects in underlying layers.
- Support for programmable shaders
  - Ability to replace the X3D lighting model with custom shader programs.
- Particle systems
  - Ability to generate systems of particles that can represent fire, smoke, and other such effects.

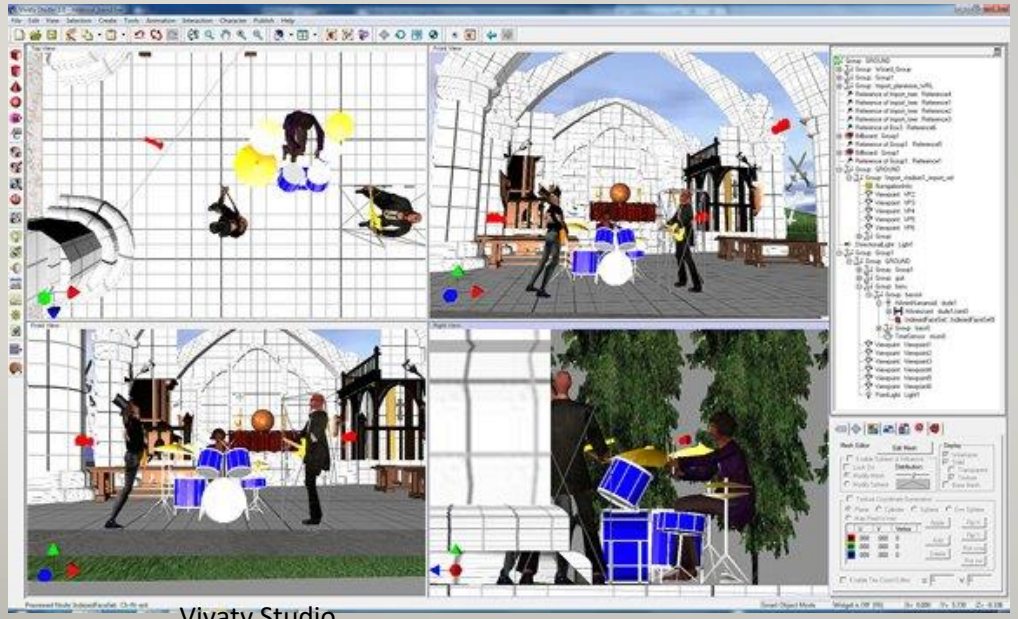
Viewing X3D files

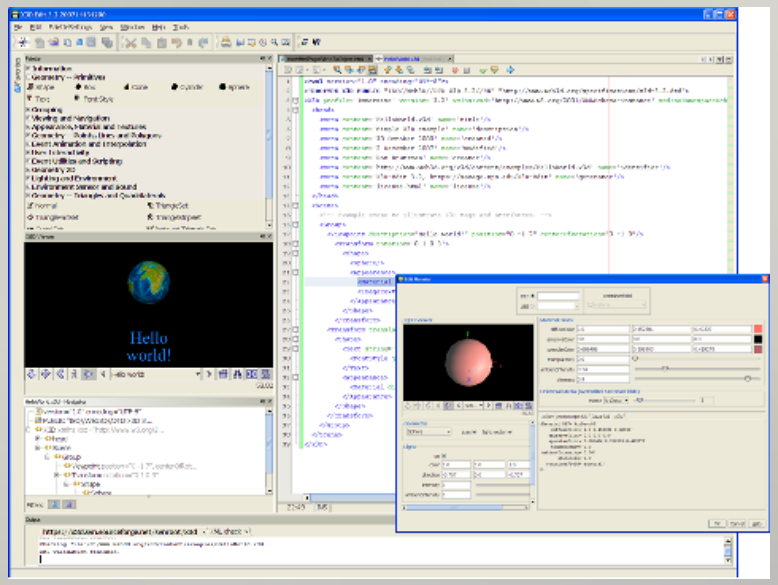
- Few current browser natively support X3D
- But there are many X3D external viewers
- BS Contact http://www.bitmanagement.de/en/download
- FreeWRL/FreeX3D http://www.crc.ca/FreeWRL
- InstantReality http://instantreality.org
- Octaga http://www.octaga.com
- Xj3D http://www.xj3d.org



Tools for X3D

- BS Editor
  - http://www.bitmanagement.de/en/products/authoring-tools/bs-editor
- X3D-Edit
  - https://savage.nps.edu/X3D-Edit/
- Vivaty Studio
  - http://www.web3d.org/realtime-3d/x3d-vrml/vivaty-studio-download

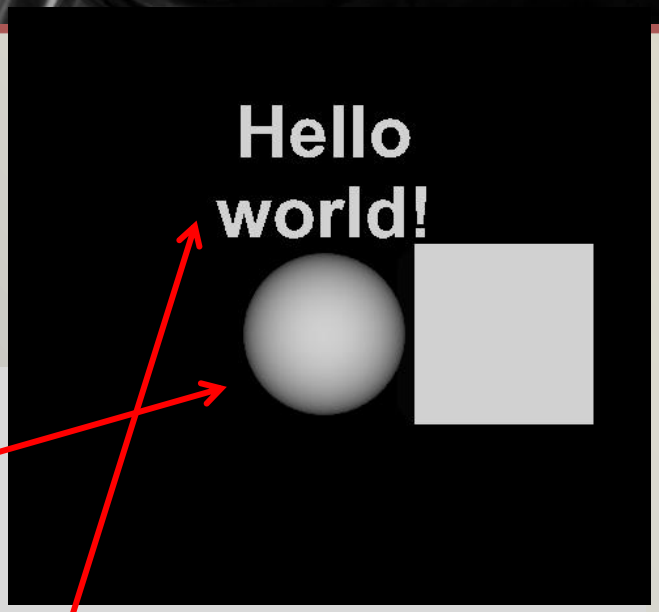




Vivaty Studio
Image courtesy of Nicholas Polys and Virgina Tech

X3D Elements

- Use tag elements to define objects, transformation, materials, textures and etc.

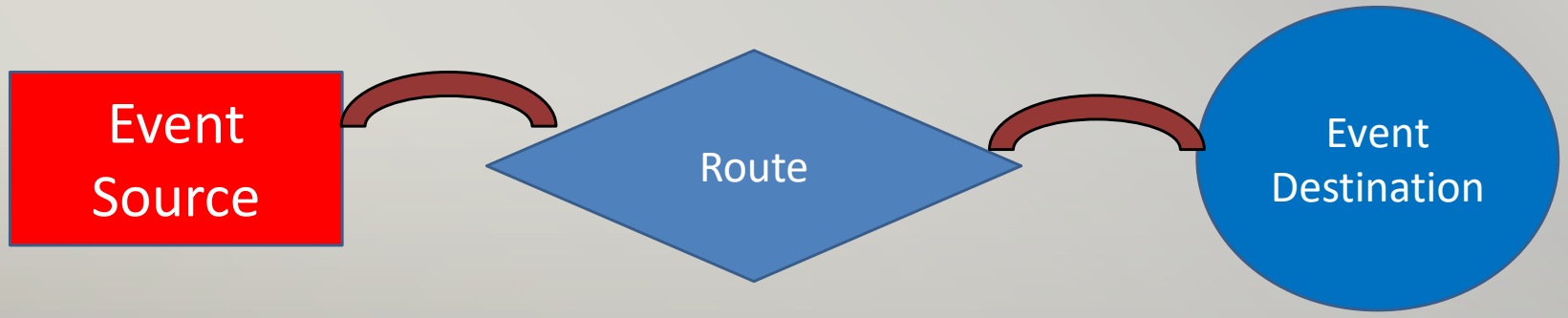



```
<Transform translation=‘2 0 0'>
        <Shape DEF='myBox'>
                <Box size='2 2 2'/>
                <Appearance>
                        <Material diffuseColor='1 1 1'/>
                </Appearance>
        </Shape>
</Transform>
<Transform translation='0 3 0'>
    <Shape DEF='myText'>
      <Text string='"Hello" "world!"' solid='false'>
      <Appearance><Material diffuseColor='1 1 1'/></Appearance>
    </Shape>
</Transform>
```

X3D Elements: Sensor Node, Route and Event

- Special mechanism is available for interactions
- Sensor is responsible to create events

- Route is to bridge between event source and event destination



Example

- E.g. Touch Sensor

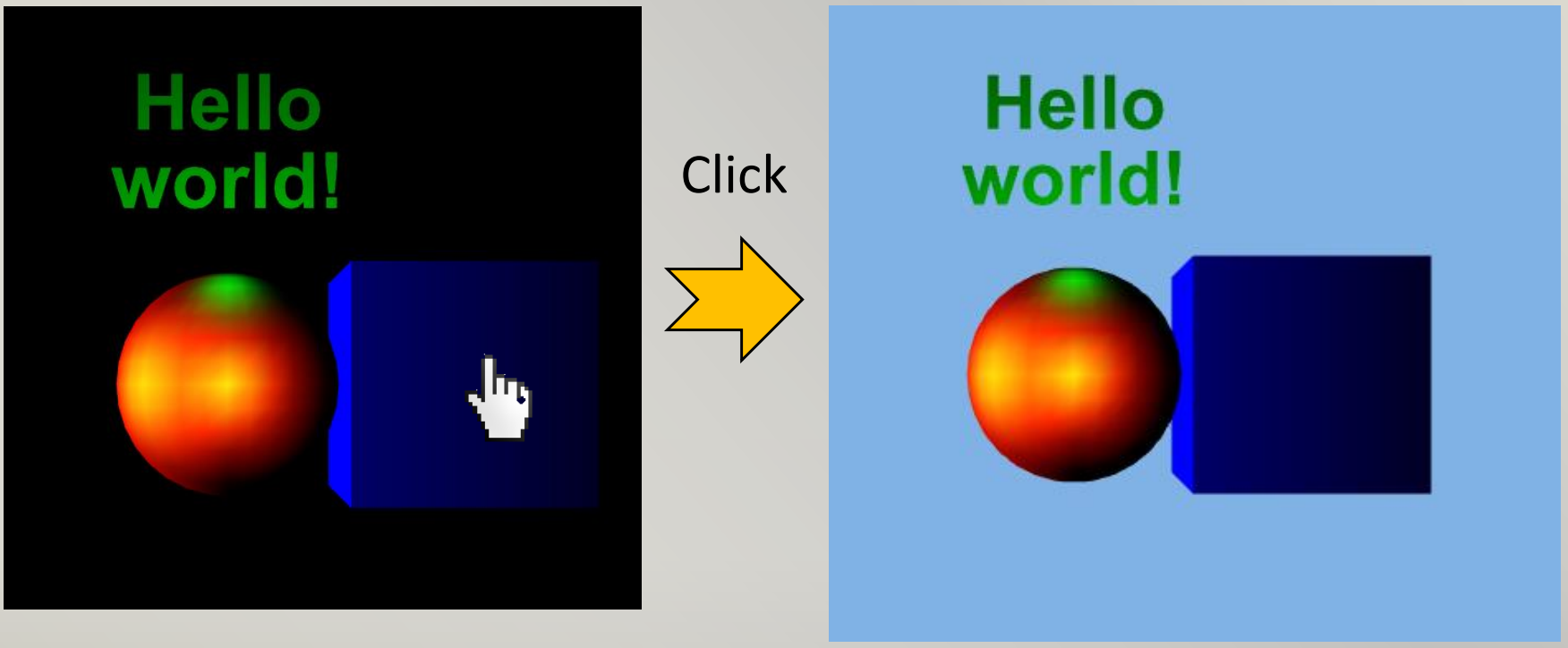
```
<Transform translation='2 0 0'>
    <TouchSensor DEF='boxTouch'/>
        <Shape DEF='myBox'><Box size='2 2 2'/>
        <Appearance>  <Material diffuseColor='0 0 1'/>
.....
</Transform>
```

- E.g. Route

```
<ROUTE fromField='isActive' fromNode='boxTouch'
toField='set_bind' toNode='BackgroundboxTouch'/>
```

Example

- Clicking a shape to change background color



Applications : X3D

- Online 3D City Navigation
- http://www.bitmanagement.de/download/BS_Contact/javascriptCheckInstall/check-install.php?url=http://www.bitmanagement.de/demos/CityShowcase/showcase.wrl&fullPage=yes

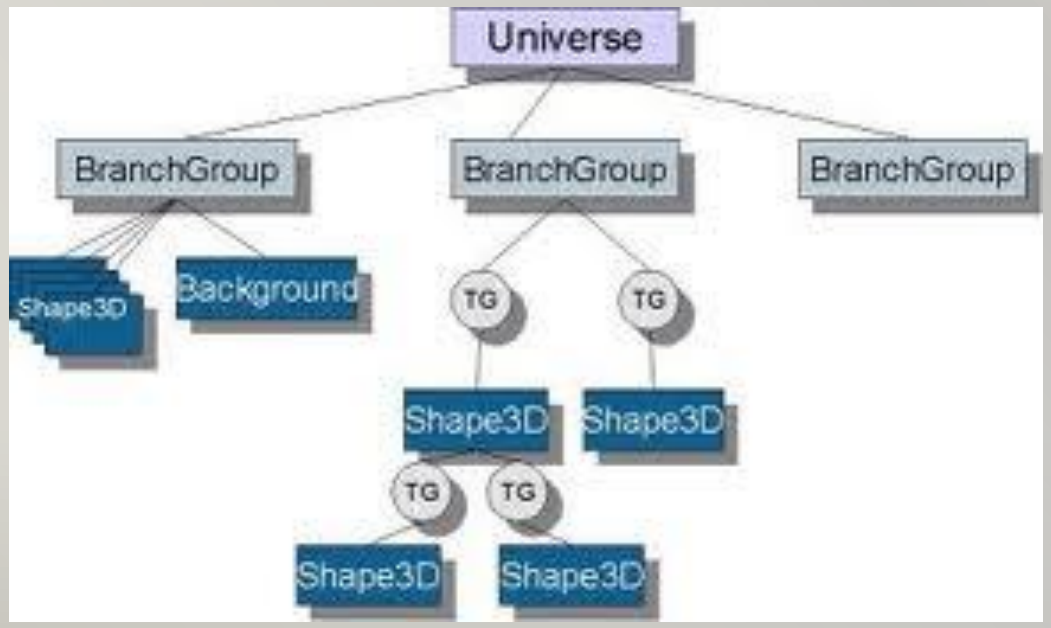
- Hand Bag Promotion
- http://www.swirlx3d.com/egbag.php

Java3D

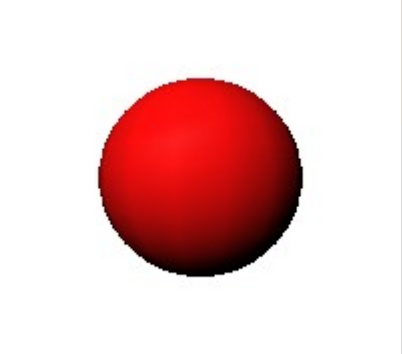
- An API/library under the family of Java
- Not specific for web applications, but can be run with use of Java Applet
- Although many browsers have JVM, many of them restrict or not recommend running Java
- Easy to learn if one is familiar with Java programming
- The application is cross platform
- Interaction or other computations can depend on existing Java APIs

Java 3D Elements

- Again, program the graphics in a scene graph paradigm
- Root of scene graph: Universe class
- Sub braches under root : Branch Group



Java 3D Elements : Example



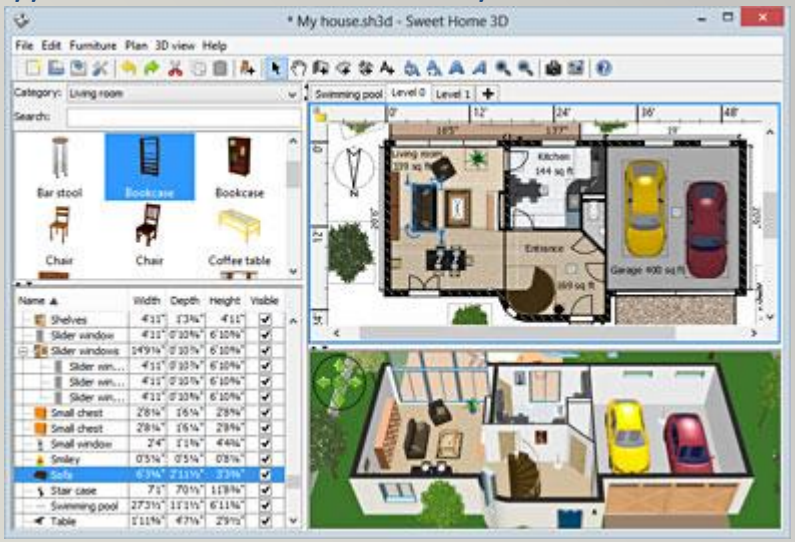
```
SimpleUniverse universe = new SimpleUniverse();
  // Create a structure to contain objects
  BranchGroup group = new BranchGroup();

  // Create a ball and add it to the group of objects
  Sphere sphere = new Sphere(0.5f);
  group.addChild(sphere);
  // Create a red light that shines for 100m from the origin
  Color3f light1Color = new Color3f(1.8f, 0.1f, 0.1f);
  BoundingSphere bounds =
  new BoundingSphere(new Point3d(0.0,0.0,0.0), 100.0);
  Vector3f light1Direction = new Vector3f(4.0f, -7.0f, -12.0f);
  DirectionalLight light1  = new DirectionalLight(light1Color, light1Direction);
  light1.setInfluencingBounds(bounds);
  group.addChild(light1);
  universe.addBranchGraph(group);
```

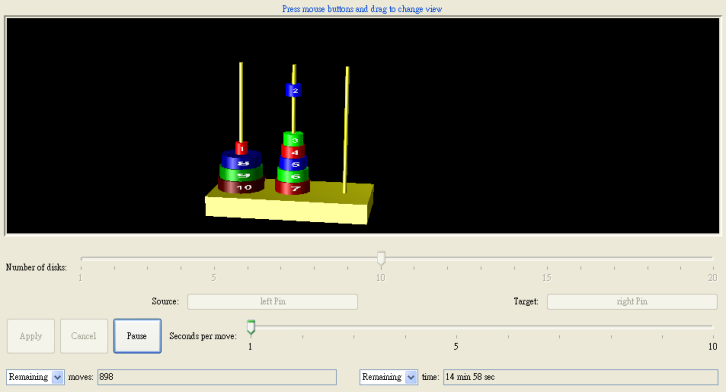
http://www.java3d.org/lighting.html

Applications : Java 3D

- Sweet Home 3D Online
  - http://www.sweethome3d.com/

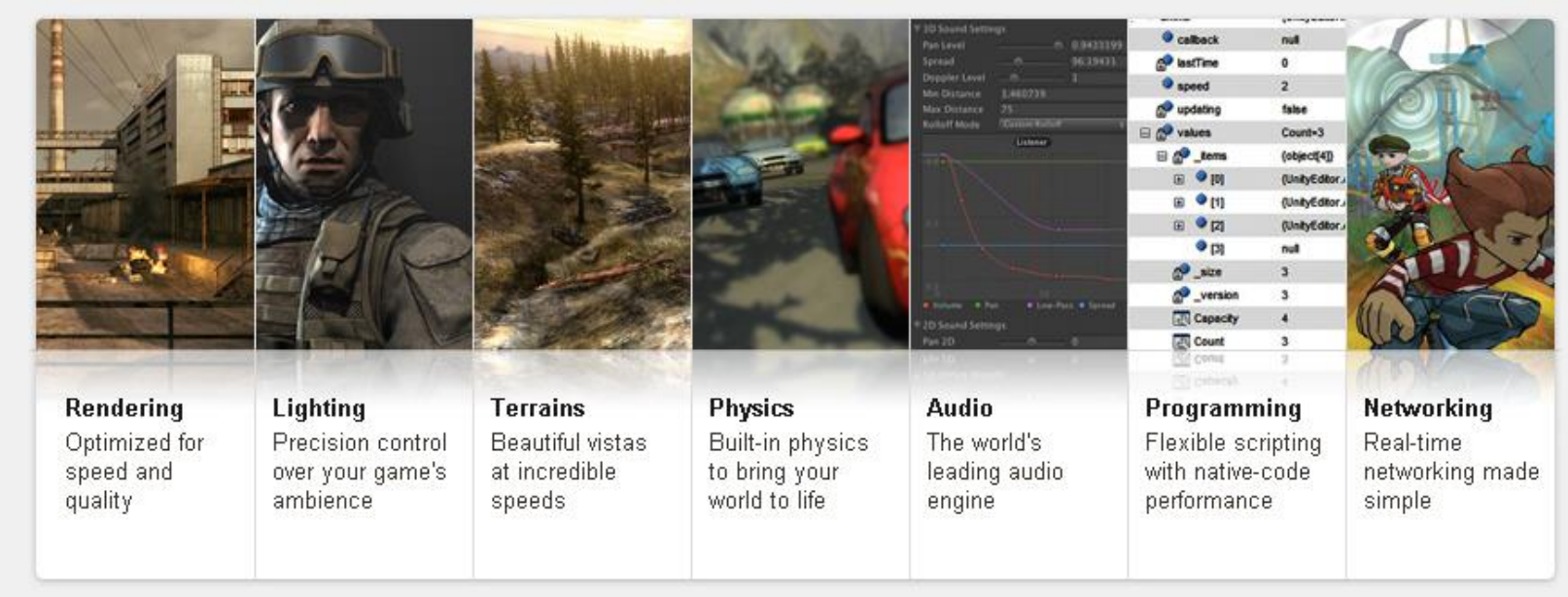


- Tower Of Hanoi (http://www.anycpu.com/projects/toh/Towers_plugin.html)



Unity3D

- A lightweight game engine
  - Not specific for web applications
- Support deployment to Web, Mobiles, PC, Consoles
- Lower learning curve

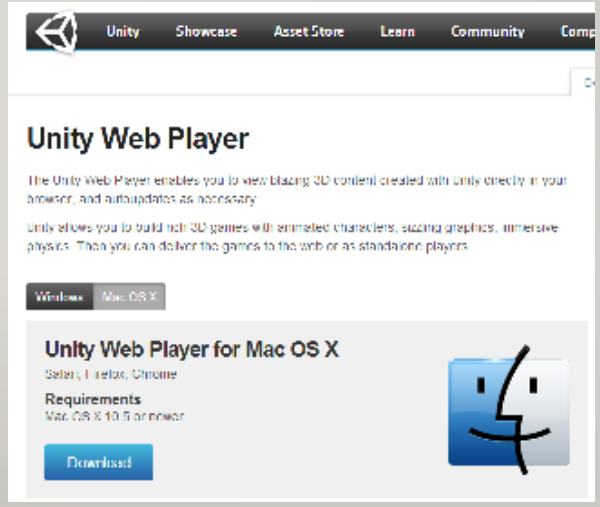


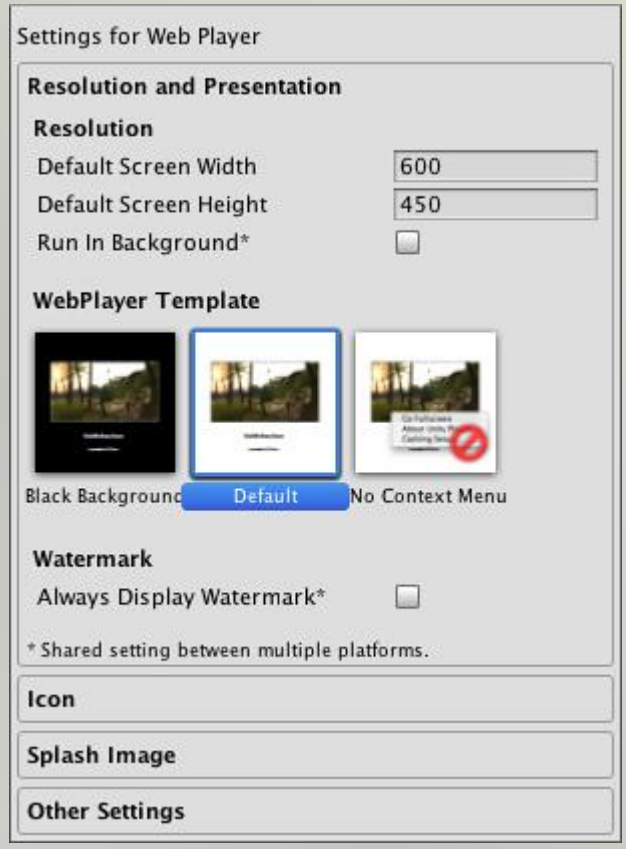
Unity3D

- Integrated editor
  - Powerful asset support
  - Blender, PS, 3DS Max, Maya, Sketchup…

Web Plugin of Unity

- Create 3D content with Unity IDE
- Build the project for Web player
- Embed the .unity3D file to web page
  - A web page can be generated by IDE with templates
- Load web page in browser with Unity Web Player

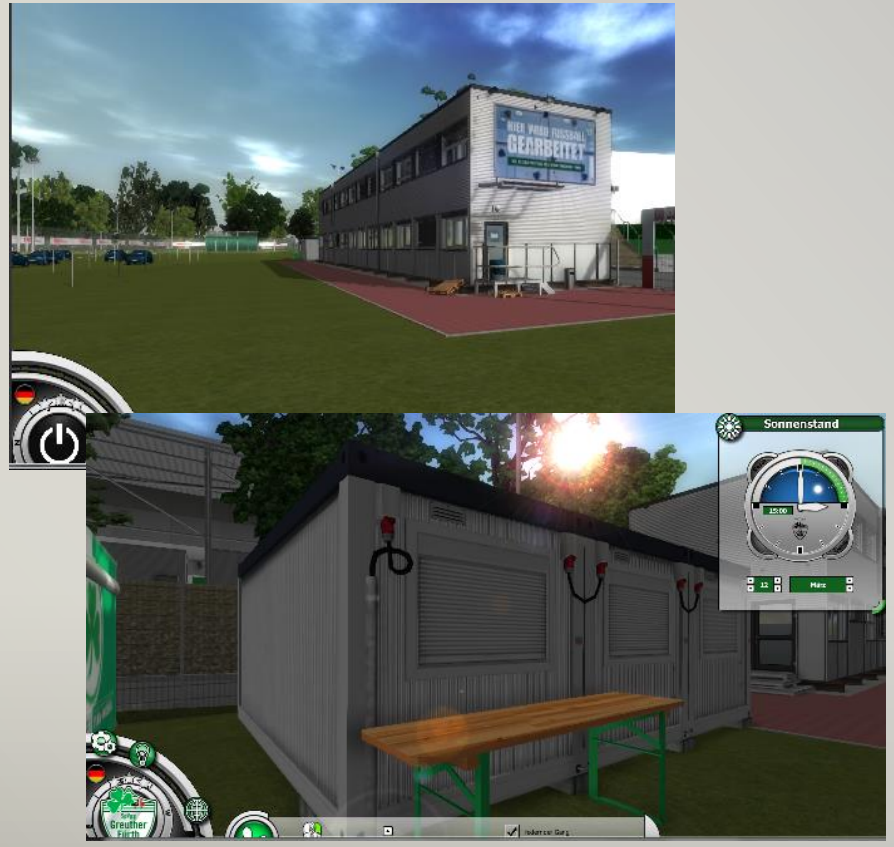




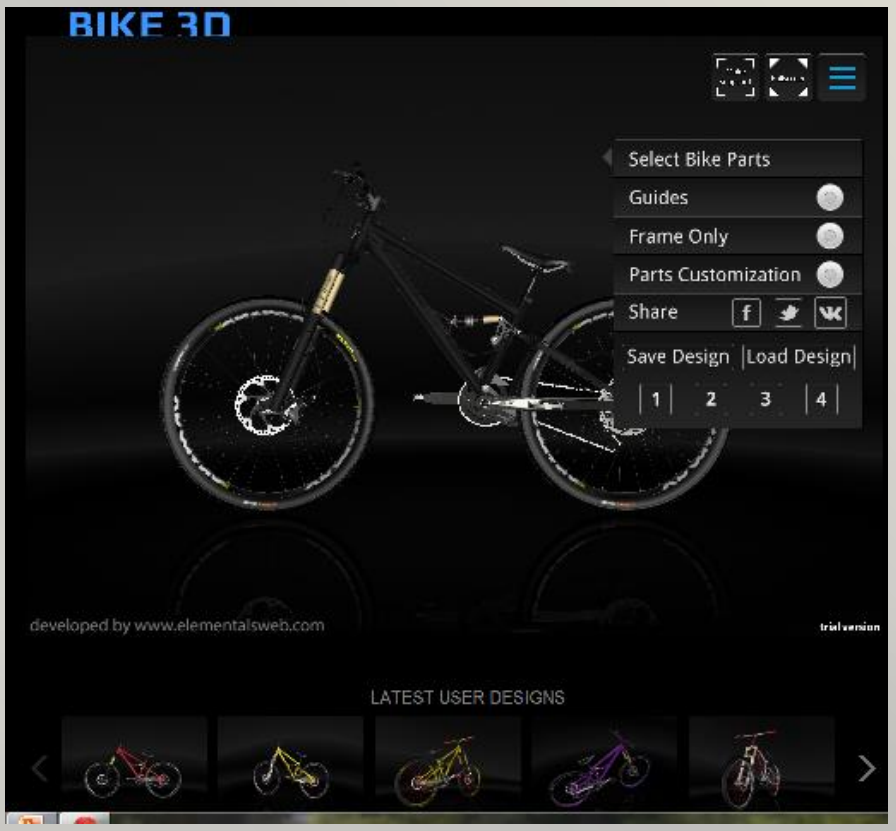
https://docs.unity3d.com/Documentation/Manual/UsingWebPlayertemplates.html

Unity3D : Applications

- 3d Stadium
  - http://www.greuther-fuerth.de/fileadmin/Inhalte/Stadiontour/

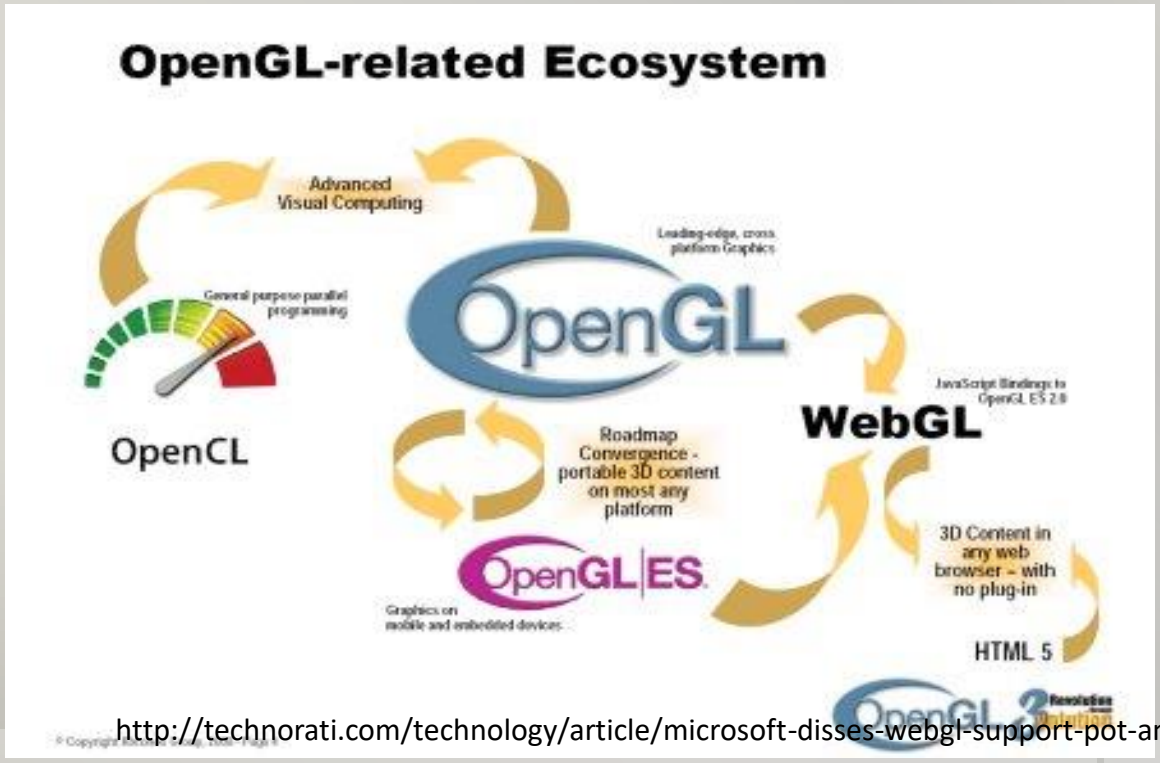


- Bike 3D configuration
  - http://www.bikeconfig.com/



WebGL

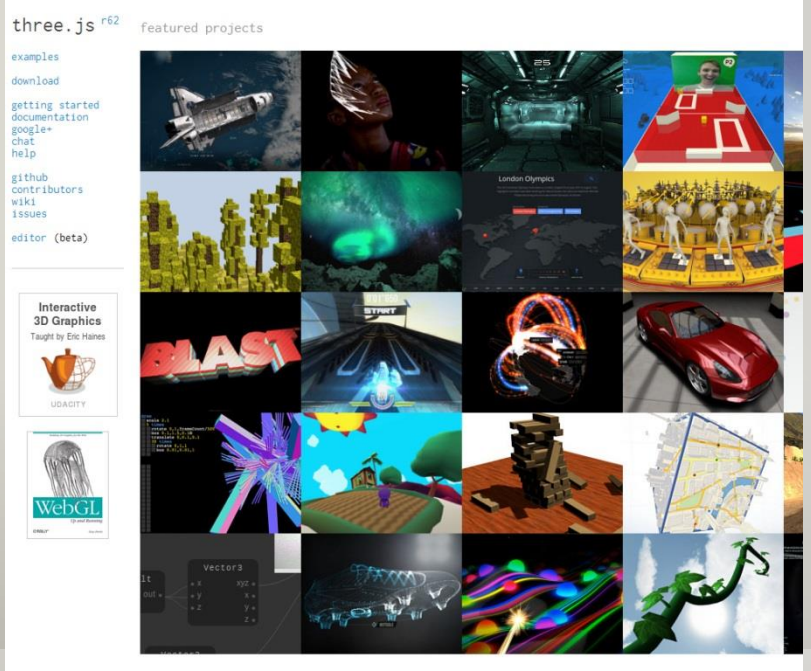
- Under the large family of OpenGL
- A low level rendering engine natively supported by most of the browsers
- Work closely with HTML5 and Javascript

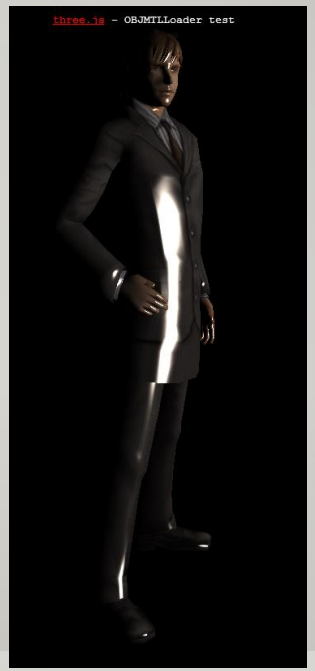


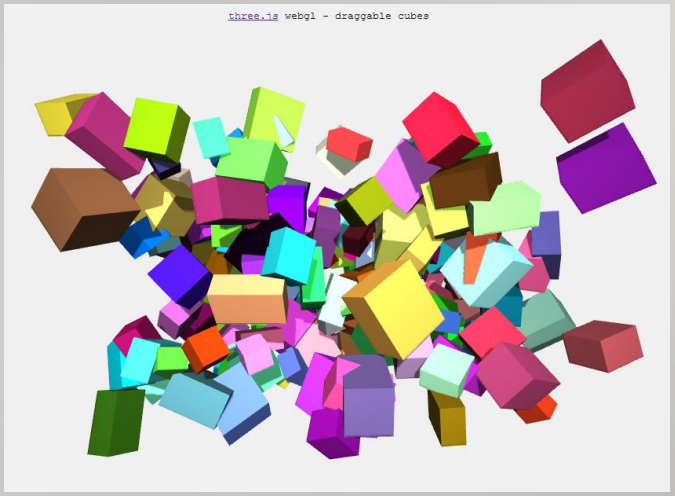
http://technorati.com/technology/article/microsoft-disses-webgl-support-pot-and/

THREEJS and WebGL

- WebGL is still a relatively new standard in the industry
- Relatively few resources to support developers
- To build prototype more easily, Threejs is higher level library built on top of WebGL
  - Refers to our tutorial notes for more details

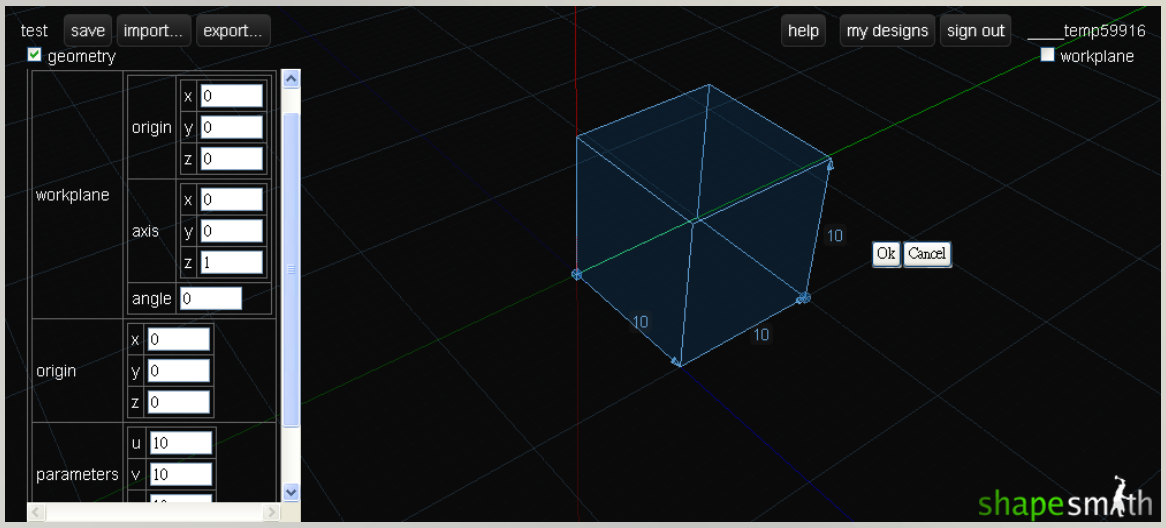






WebGL: Applications

- Shapesmith
  - http://shapesmith.net/



- From Dust (Game)
  - http://from-dust.ubisoft.com/

Augmented Reality

Augmented Reality : the Definition

- Critical Characteristics [Azuma97]
  - Combines Real and virtual objects in the same scene
  - Interactive in real-time

    Can interact with virtual objects
  - Registered in 3D

    Virtual contents appear fixed in space

The Times of London: Spots to accompany a piece on augmented reality (AR) on the web and in baseball cards

Azuma, R. T. (1997). A survey of augmented reality, Presence 6(4), 355-385.

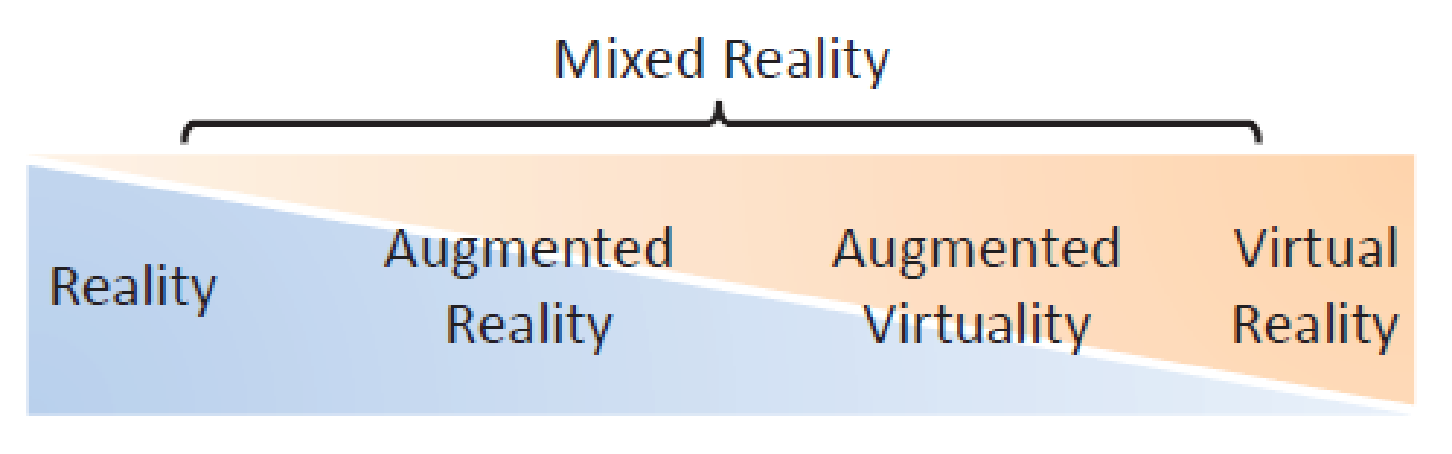
VR and AR

- Virtual Reality: Replaces Reality
  - Scene Generation: requires realistic images
  - Display Device: fully immersive, wide FOV
  - Tracking and Sensing: low accuracy is okay
- Augmented Reality: Enhances Reality
  - Scene Generation: minimal rendering okay
  - Display Device: non-immersive, small FOV
  - Tracking and Sensing: high accuracy needed

From Mark Billinghurst

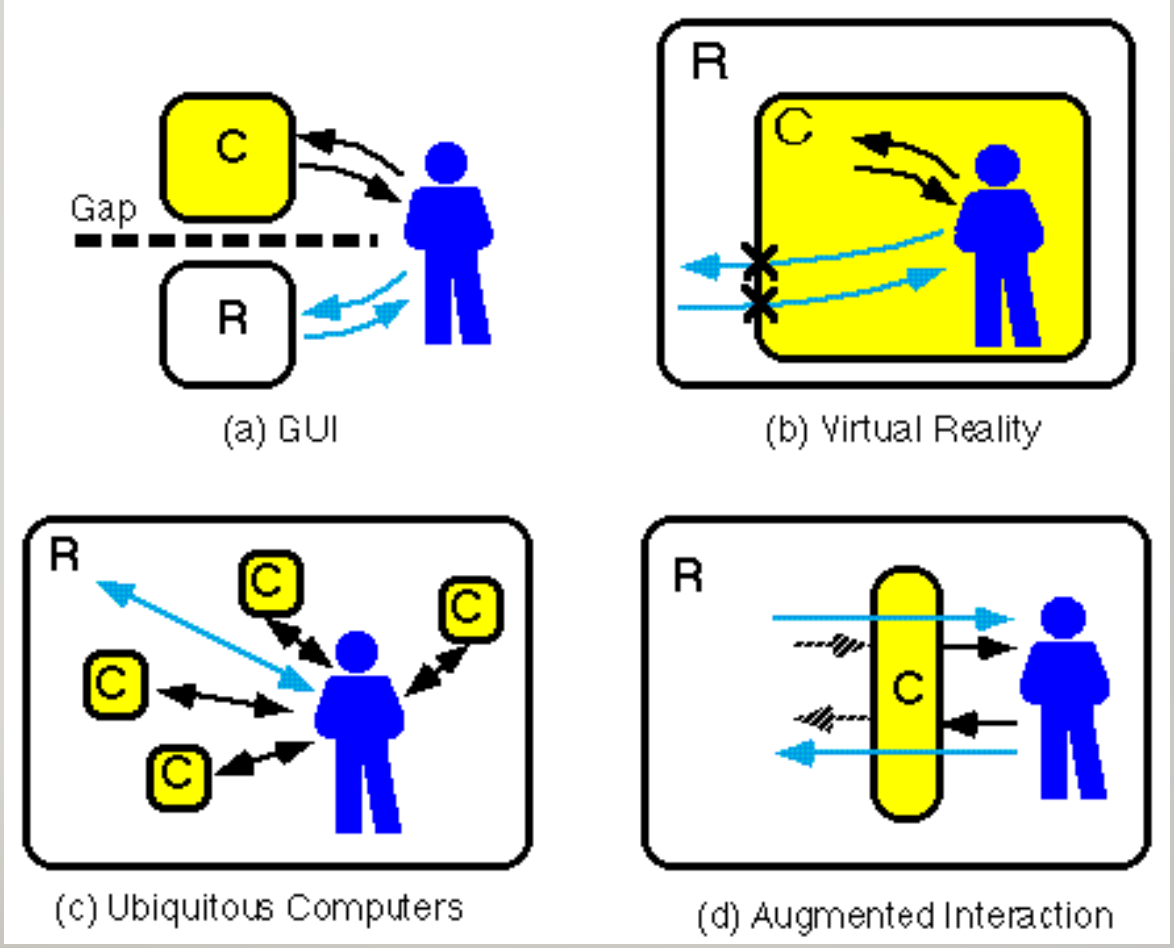
Mixed Reality (MR)

- The space between reality and virtual reality, which allows real and virtual elements to be combined to varying degrees, is called **mixed reality**.



Interfaces and Interaction

- C: Computer , R: Reality



Examples

- Overlaying virtual objects / contents on camera view

Examples

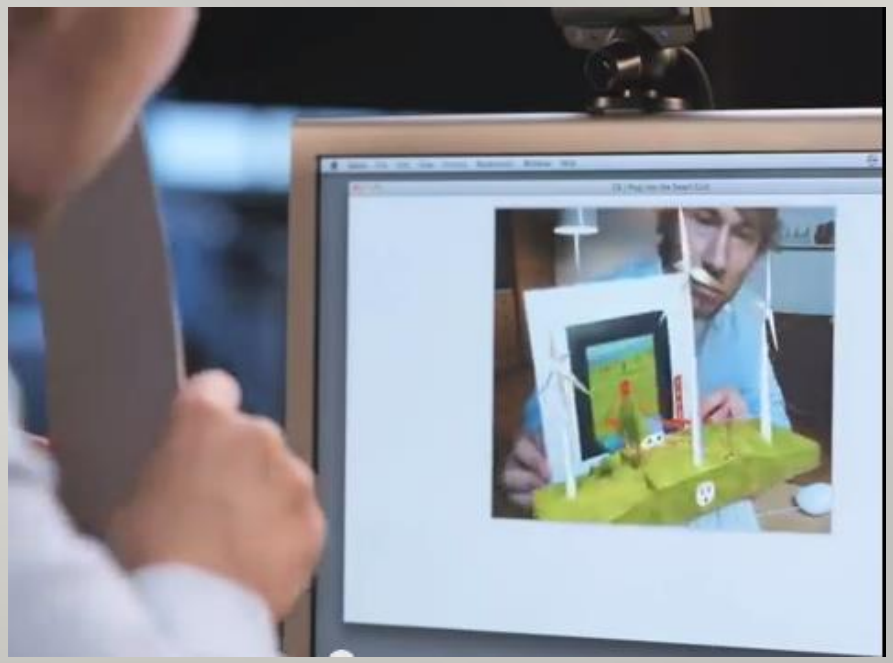
- Mini-tanks came out in sequence
  - http://www.youtube.com/watch?v=oiqIPXnKkKo#t=29
- AR story book
  - http://www.youtube.com/watch?v=1qHh3hVcu5U
- 3d video for "Ride My Star"
  - http://www.youtube.com/watch?v=Z_bj1GTH9UA#t=24

Web-based AR

- Based on Browser
- Implemented with Flash + web camera and 3D graphics
- Popular after web camera is a standard device in PC

Example

- GE using FLARToolKit in the Smart Grid
  - http://www.youtube.com/watch?v=NK59Beq0Sew



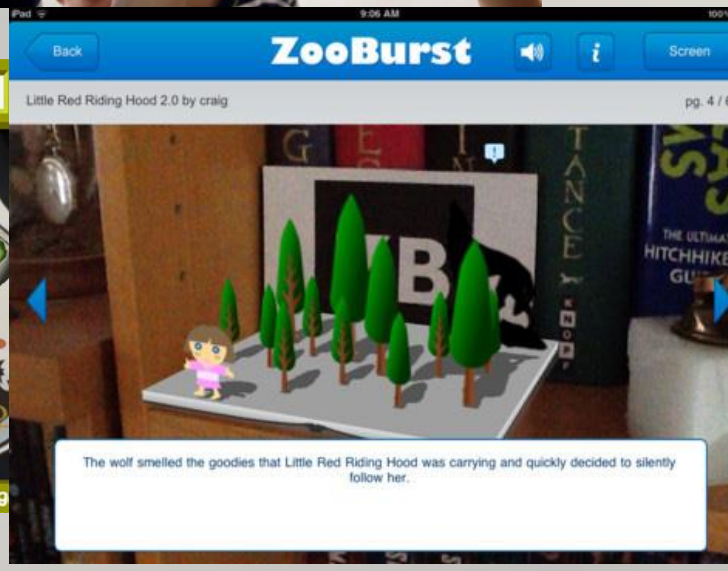
Mobile AR

- Rely on various sensors in a mobile device
- GPS + compass
  - Discover Hong Kong AR
  - Google Sky Map
  - Acrossair
  - Layar
  - PressLite
  - Yelp
  - AR Car Finder
  - Etc.

Discover Hong Kong·AR
https://itunes.apple.com/app/id437814235?mt=8

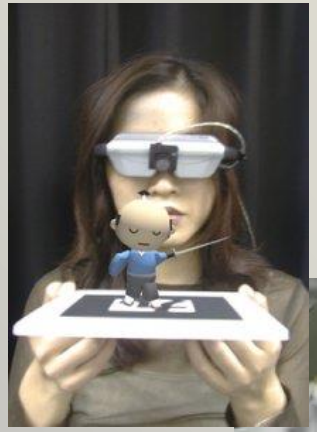
Mobile AR

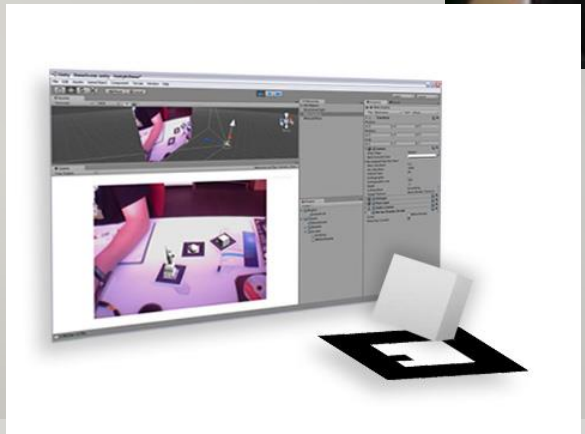
- Vision-based
  - IKEA AR Furniture
  - AR Basketball
  - FETCH! Lunch Rush
  - ZooBurst



Tools for AR Development

- Low Level Software Libraries
  - ARToolkit
  - osgART (ARToolkit with OpenSceneGraph)
  - MXRToolKit, Etc
- Plug-ins to existing software
  - DART (Macromedia Director),
  - mARx,
  - Unity3D

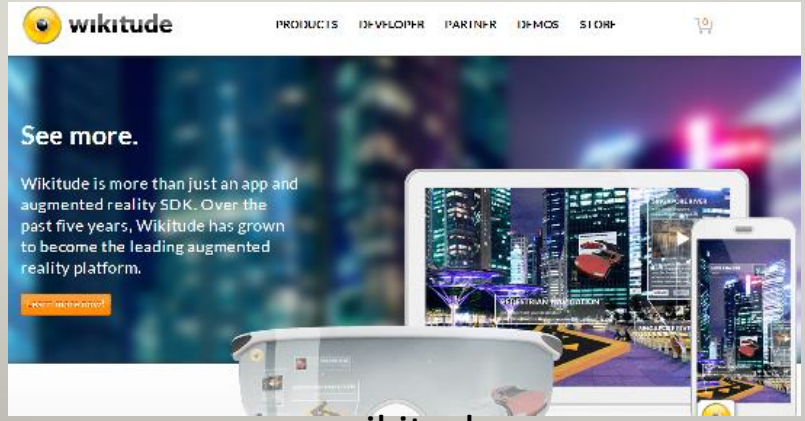




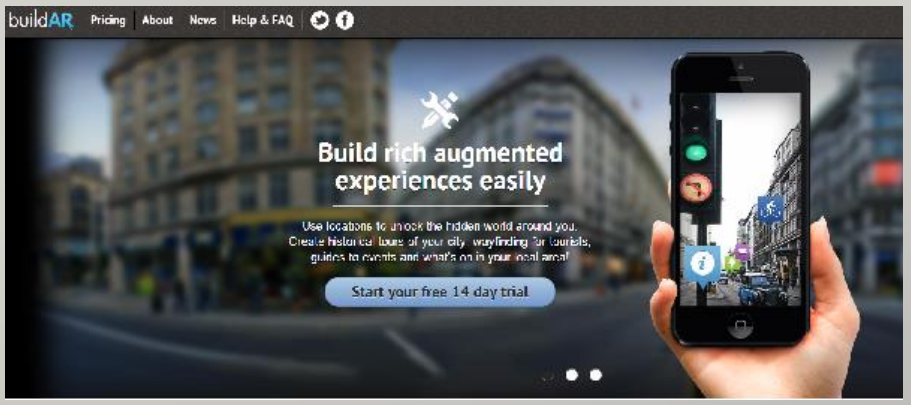
Tools for AR Development (cont')

- Stand Alone
  - AMIRE (free)
  - BuildAR
  - Metaio Creator
  - wikitude (mobile)

AMIRE



wikitude



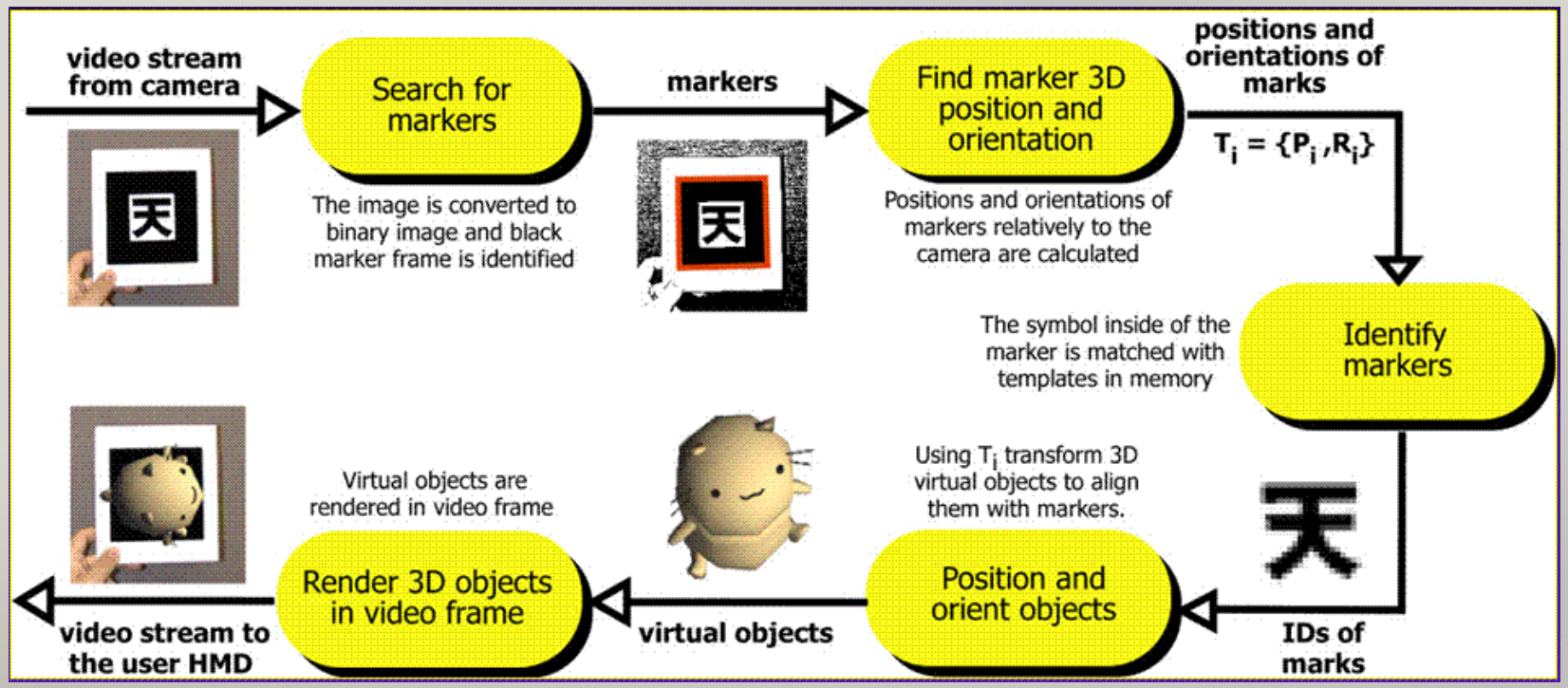
BuildAR

ARToolkit Development

- Cross platform
  - Windows, Mac, Linux, IRIX, Symbian, iPhone, etc
- Additional basic libraries
  - Video capture library (Video4Linux, VisionSDK)
  - OpenGL, GLUT
- Standard ARToolKit just provides tracking functionality
- For an AR application, you'll need high level rendering library and/or Audio Library
  - Open VRML, Open Inventor, osgART, etc
  - Fmod, etc

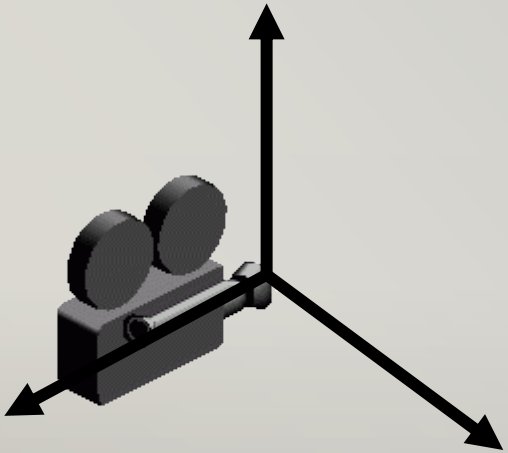
ARToolkit [Kato 1998]

- Marker based AR
- Open source – computer vision based AR tracking
- http://artoolkit.sourceforge.net/

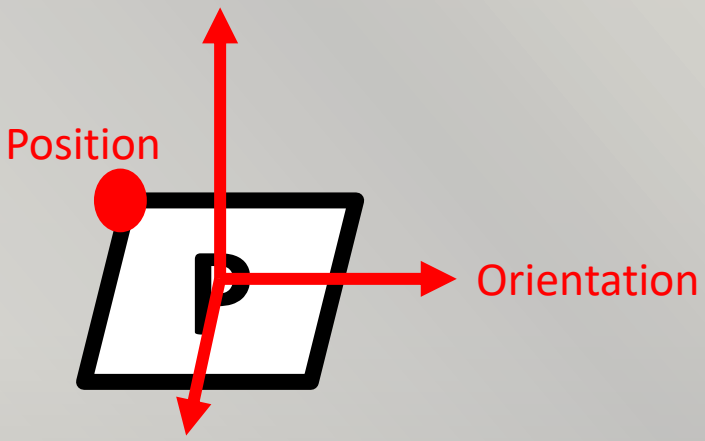


What had been computed by ARToolkit?

- Position of makers in the camera coordinates
- Pose of markers in the camera coordinates



Camera Space



Marker

ARTookit Implementation

- Initialization
  - Load camera and pattern parameters
- Main Loop

Step1. Image capture and display

Step2. Marker detection

Step3. Marker identification

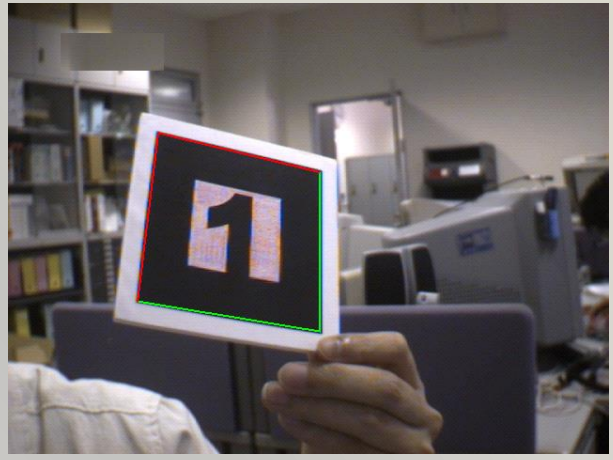
Step4. Getting pose information

Step5. Object Interactions/Simulation

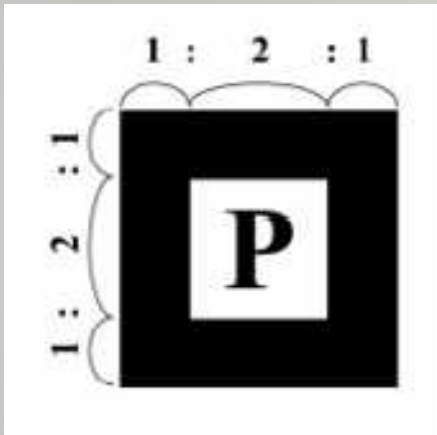
Step6. Display virtual objects

Marker Template and Recognition

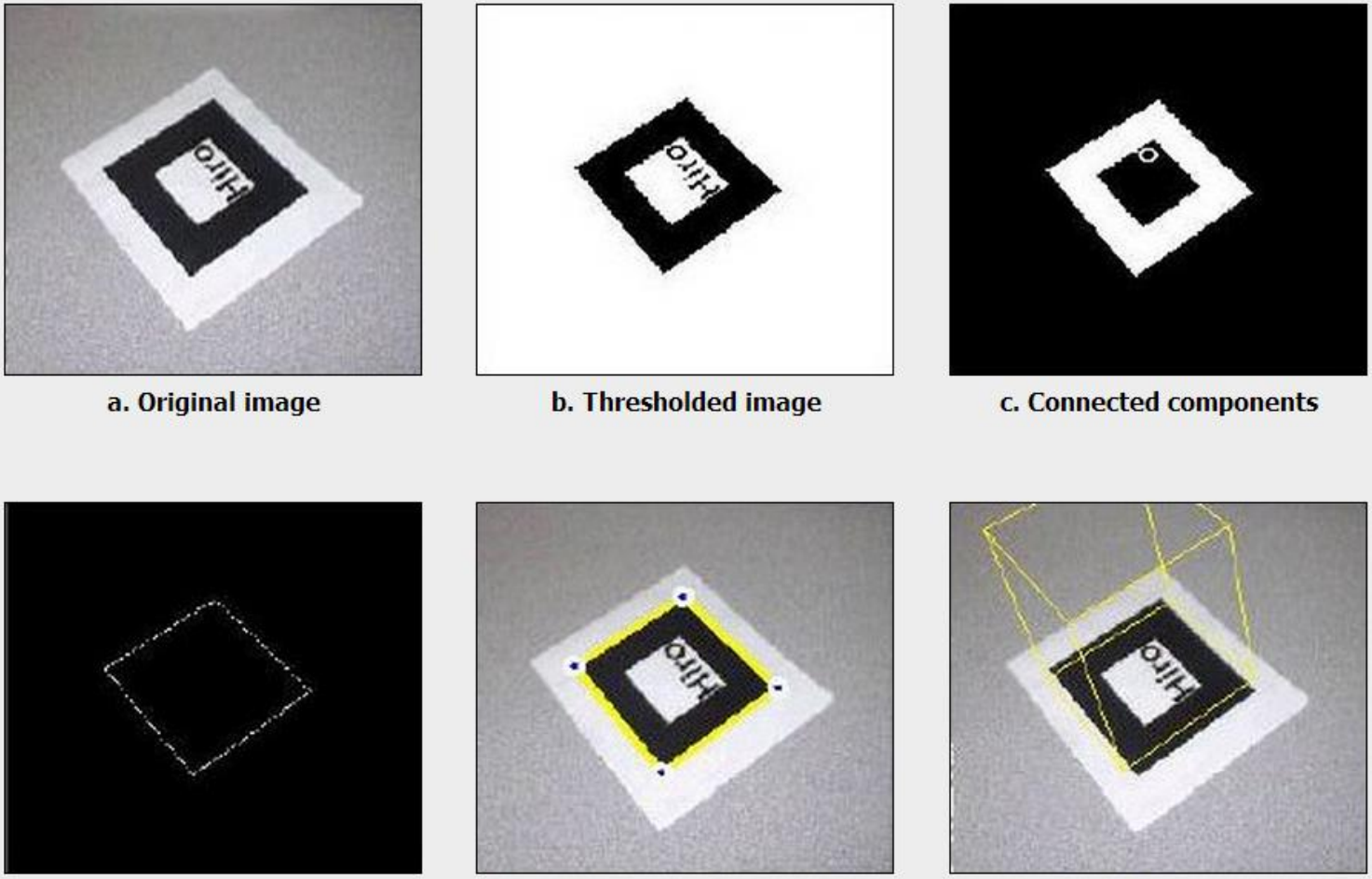
- Preparing the marker
  - Generate using utility program (mk_patt.exe)
  - 1:2:1 ratio determines the pattern region used
  - The border is used for marker detection, must be in pure black



- Marker Recognition involves
  - Basic image processing to enhance contrast
  - Edge and corner detection
  - Fast pose estimation algorithm



Marker Recognition



a. Original image

b. Thresholded image

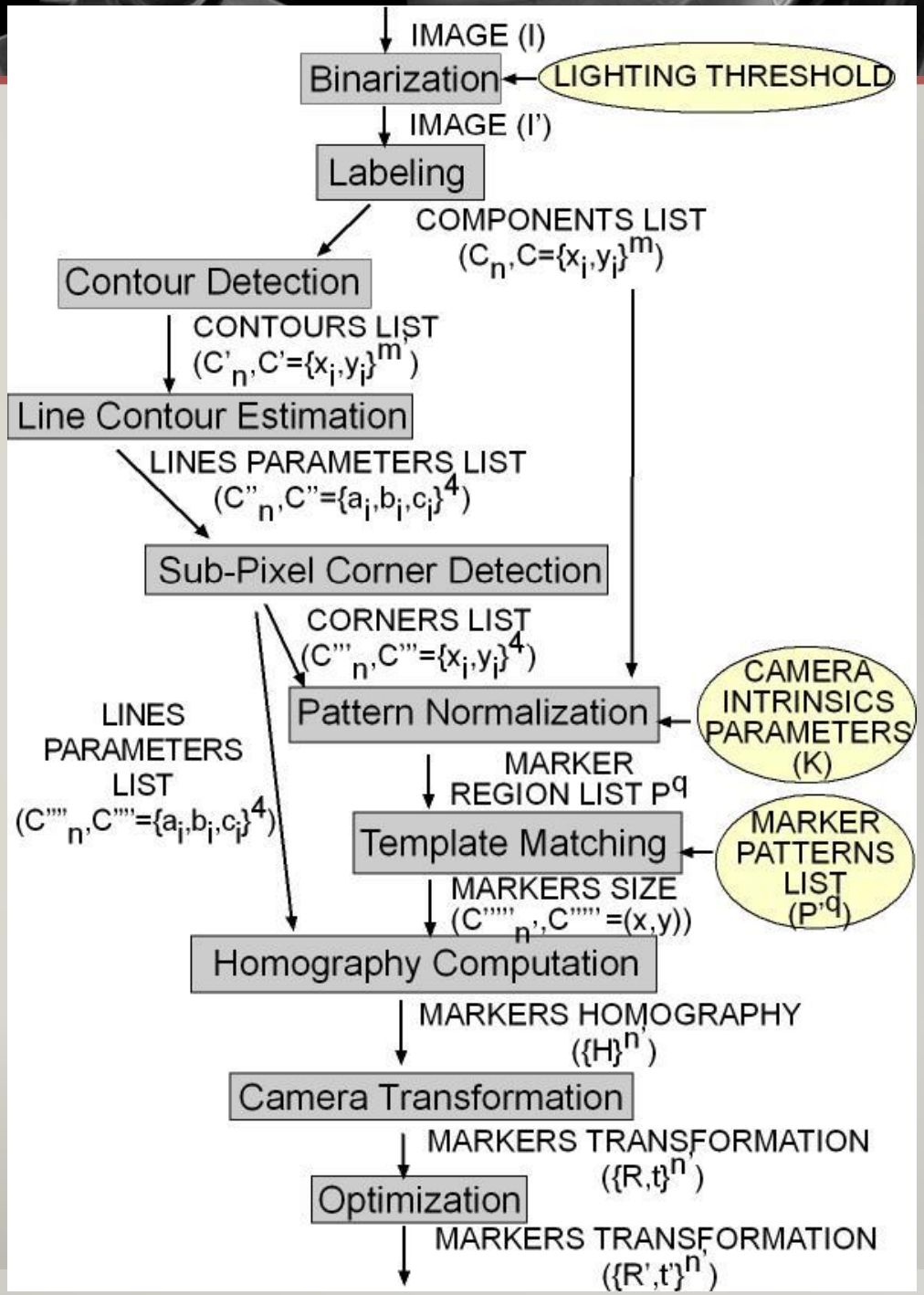
c. Connected components

d. Contours

e. Extracted marker edges and corners

f. Fitted square

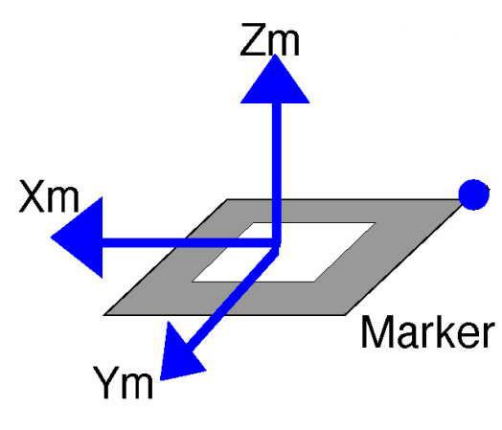
http://www.hitl.washington.edu/artoolkit/documentation/vision.htm



Major Steps

- Contour extraction
- Corner and edge detection
- Template matching
- Marker transformation estimation and optimization

Pose and Location of Marker

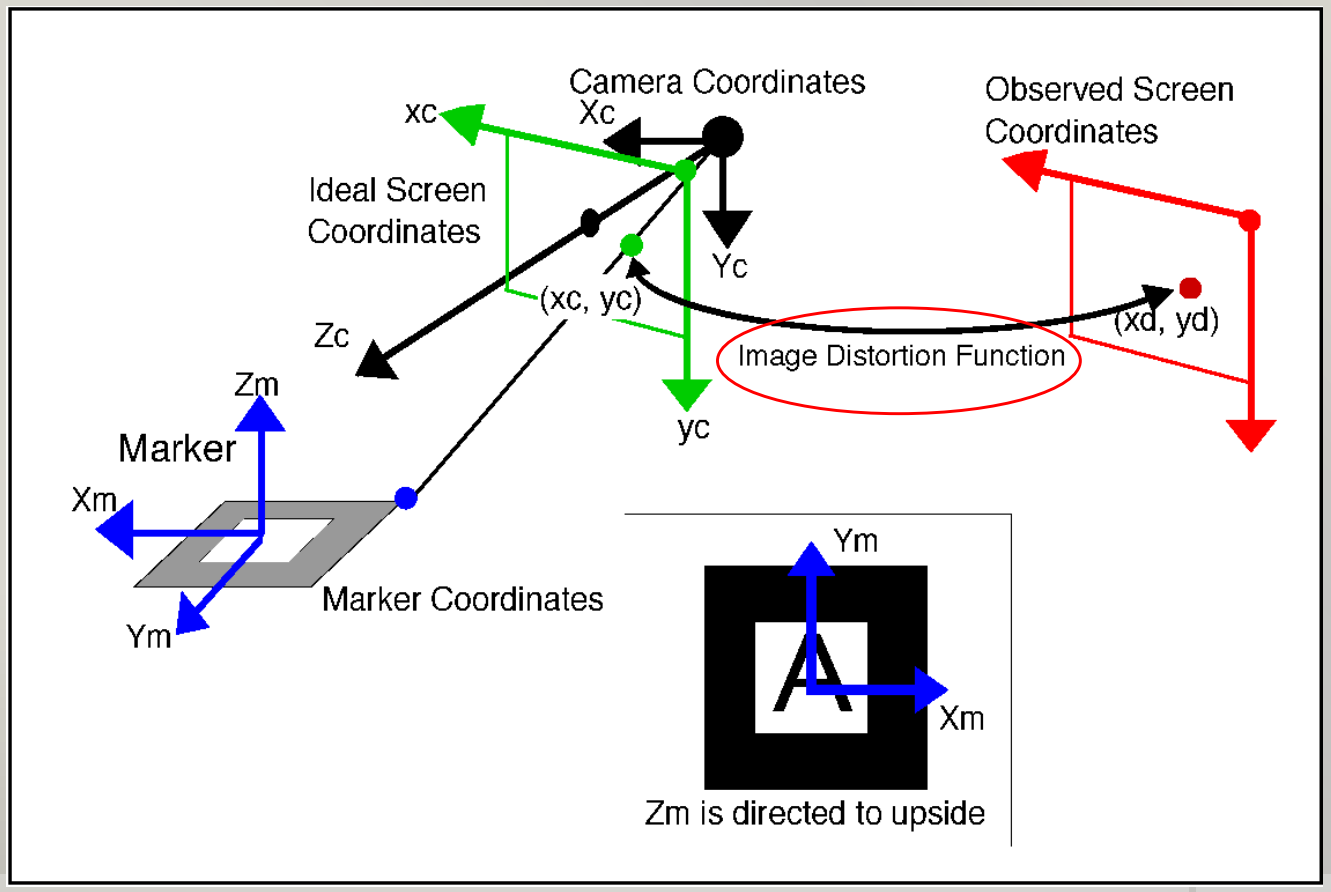


- Rotation & Translation (Orientation & Pose)
- **TCM** : 4x4 transformation matrix
- From marker space to camera space

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} = \mathbf{T_{CM}} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix}$$

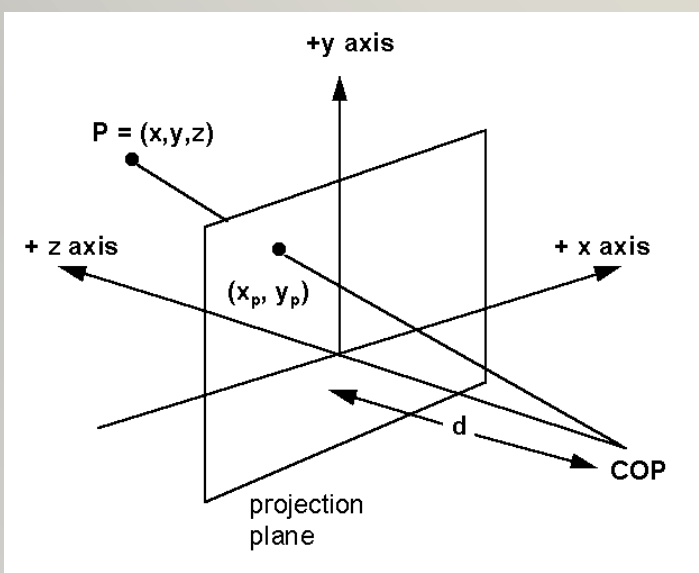
Pose and Location of Marker

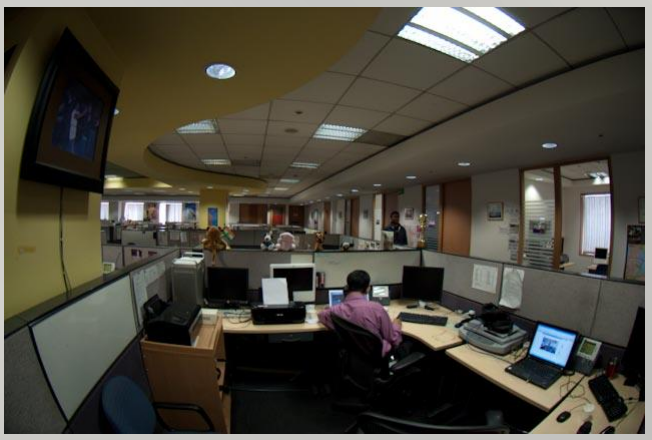
- Relationship between marker and camera
- Captured image will be distorted because of imperfect lens



Camera Configuration & Lens Distortion

- Camera configuration/ intrinsic parameters
  - Defined by the projection matrix
  - Lens distortion
- Commonly obtained by camera calibration
- Lens distortion is common in most of the cameras, most obvious at the edges

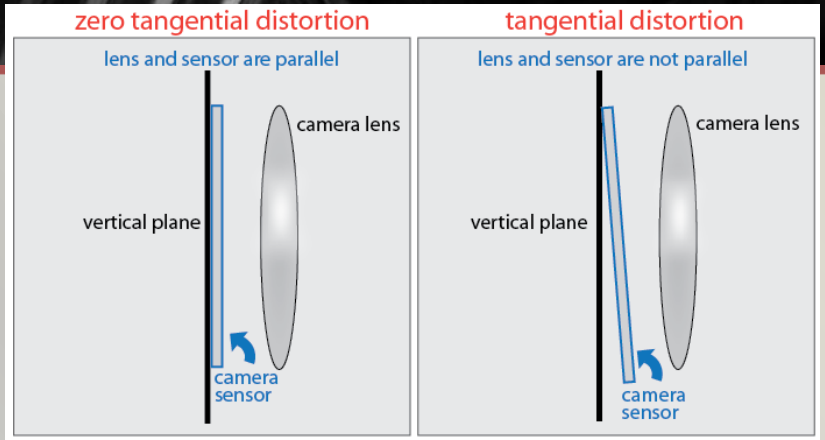


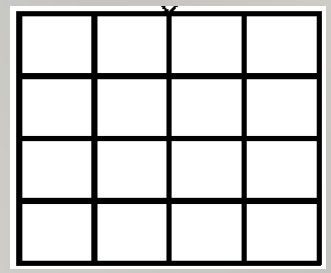


Distorted image captured from a typical camera

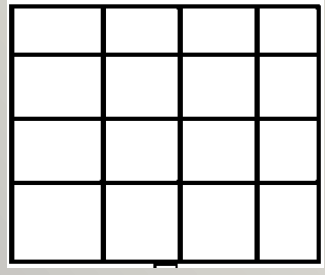
Lens Distortion

- Most lens suffer from
  - Radial, and
  - Tangential distortions
- Mainly two types of radial distortion
  - Pincushion
  - Barrel

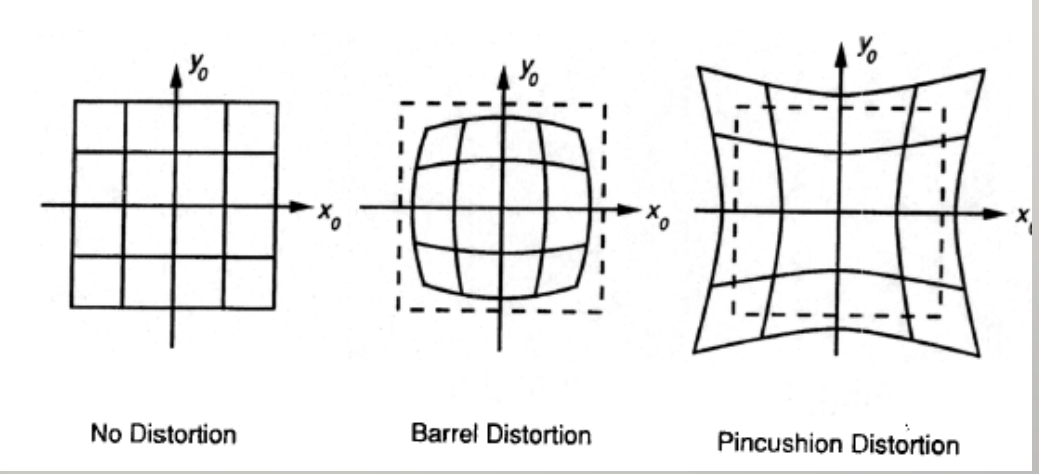




No distortion



Tangential distorted



Lens Distortion

- Understand the degree of distortion will help to rectify the distorted image
  - xu and yu are undistorted coordinates of a point with coordinates (xd, yd); radial distortions k_1,k_2 and two coefficients of tangential distortions p_1, p_2

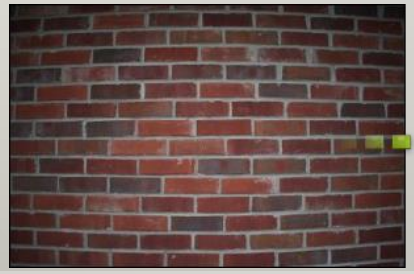
$$x_u = x_d \cdot \left(1 + k_1 r^2 + k_2 r^4\right) + 2p_1 x_d y_d + p_2 \cdot \left(r^2 + 2x_d^2\right)$$

$$y_u = y_d \cdot \left(1 + k_1 r^2 + k_2 r^4\right) + 2p_2 x_d y_d + p_1 \cdot \left(r^2 + 2y_d^2\right)$$

http://software.intel.com/sites/products/documentation/hpc/ipp/ippi/ippi_ch14/ch14_correction_of_camera_lens_distortion.html

- Try to unwarp the image, somehow similar to a mapping of 2D position

Distorted



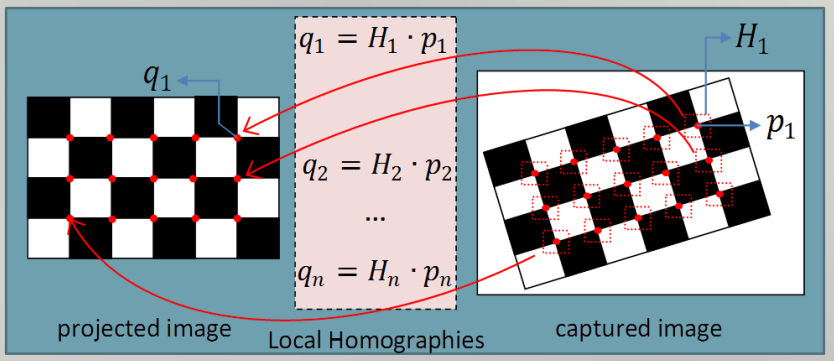
Rectified

Camera Calibration

- A classical topic in computer vision
- Fit the behavior of the camera to our pinhole camera model
- Therefore, result is a projection matrix
- Basic idea / steps of calibration
  - Use a known geometry

    Commonly a planar object with checker pattern
  - Aim the camera at the geometry in different pose
  - Corner / edge detect the known pattern
  - Solve for the projection matrix with the minimum error

Camera calibration with checker board



http://mesh.brown.edu/calibration/

Virtual Object Rendering

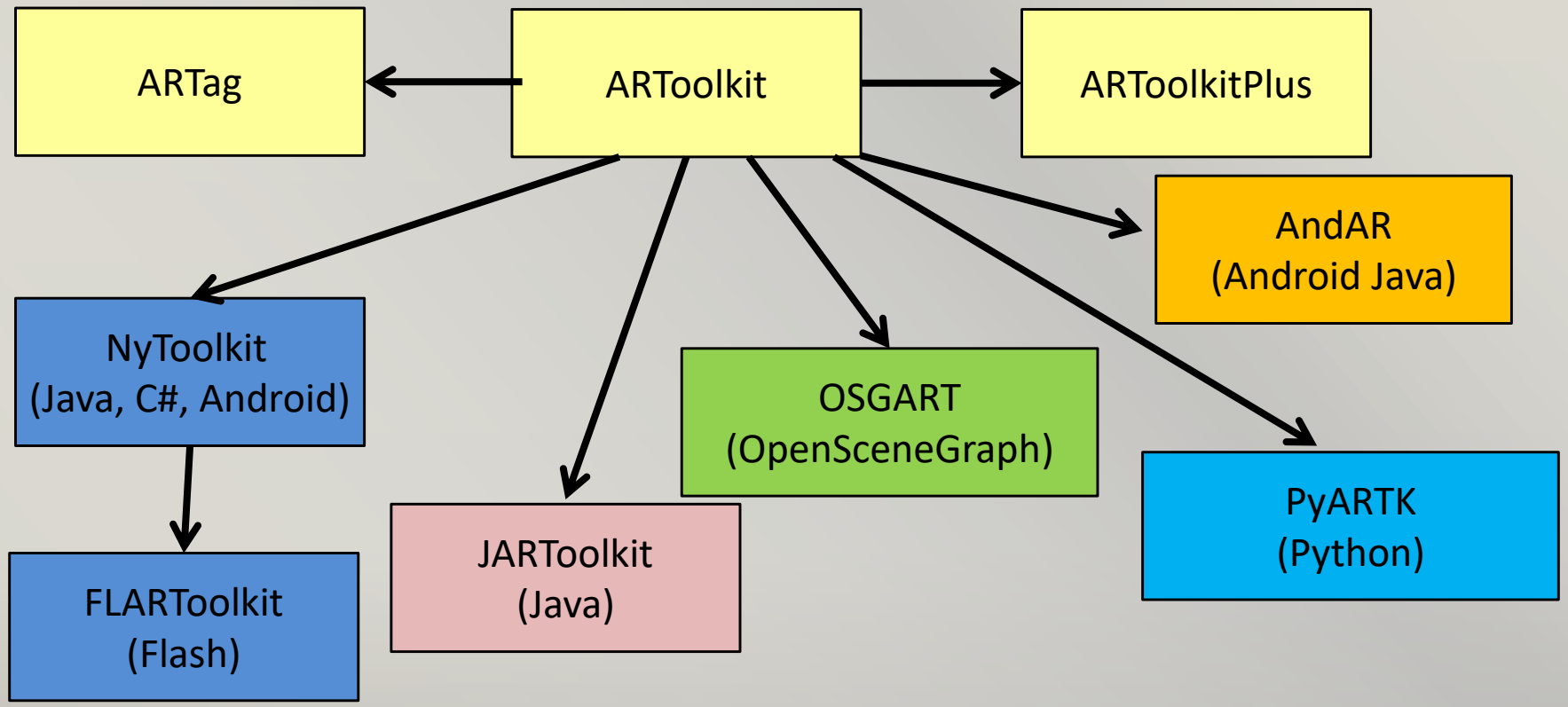
- Paint the image as background
- Set the camera parameters to **OpenGL Projection** matrix.
- Set **OpenGL ModelView** matrix as the transformation matrix from the marker to the camera.
- Virtual objects will be drawn above the marker

ARToolKit Applications

- Hundreds of projects in research community

The Family of ARToolkit

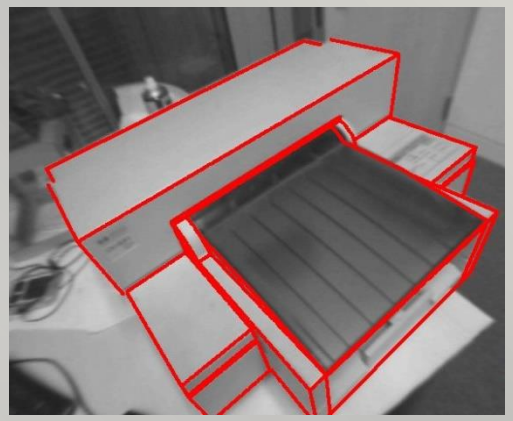
- There are a family of projects / language bindings derived from the standard ARToolkit



Why SLAM

- Traditional approach based on known map (/geometry)
  - Fiducial marker
  - 3D CAD models

- Is it possible if no such assumption?
- i.e. Track without prior model of world



- Challenges:
  - Speed
  - Accuracy
  - Robustness
  - Interaction with real world

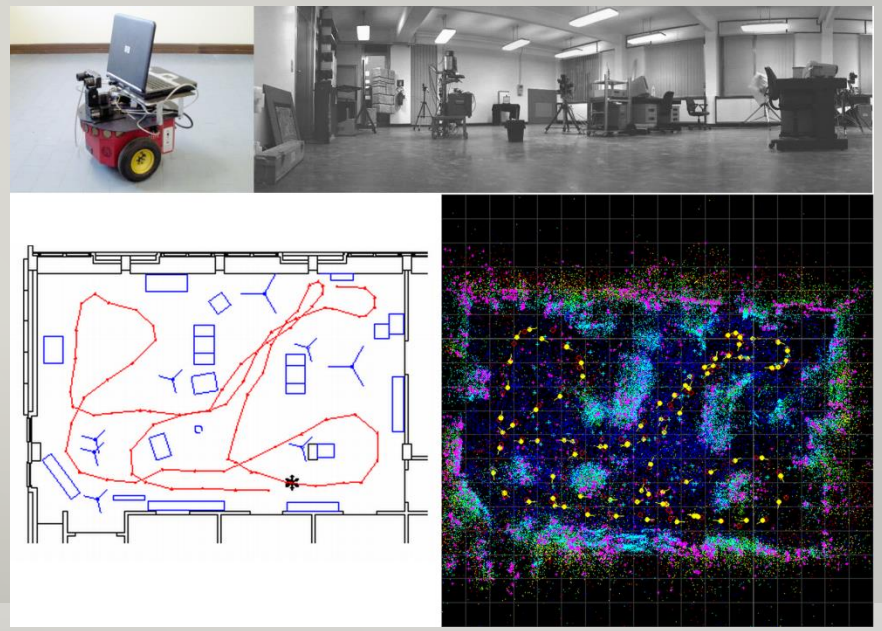
The SLAM Problem

- SLAM (Simultaneous Localization And Mapping)
  - A research area in robotics
- **Localization:** inferring location given a map
- **Mapping:** inferring a map given a location
- **SLAM:** learning a map and locating the robot simultaneously

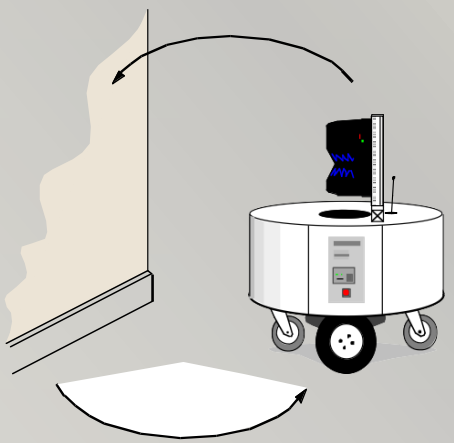
SLAM is the process by which a robot **builds a map** of the environment and, at the same time, uses this map to **compute its location**

The SLAM Problem

- SLAM is considered one of the most fundamental problems for robots to become truly autonomous
- Commonly involve
  - Robot's control or motion
  - Use a rich array of sensors - lasers, sonar, MWR, gyros & cameras



The SLAM Problem



- SLAM is a **chicken-or-egg problem**:
  - → A map is needed for localizing a robot
  - → A pose estimate is needed to build a map
- Thus, SLAM is (regarded as) a **hard problem** in robotics

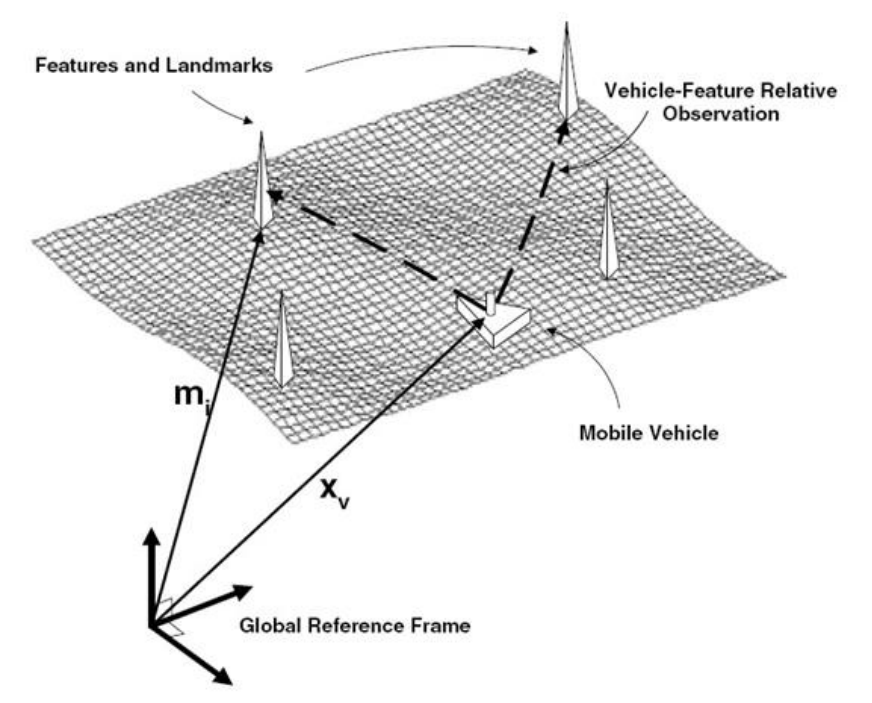
The SLAM Problem

Given:

- The robot's controls
  $\mathbf{U}_{0:k} = \{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\}$
- Relative observations

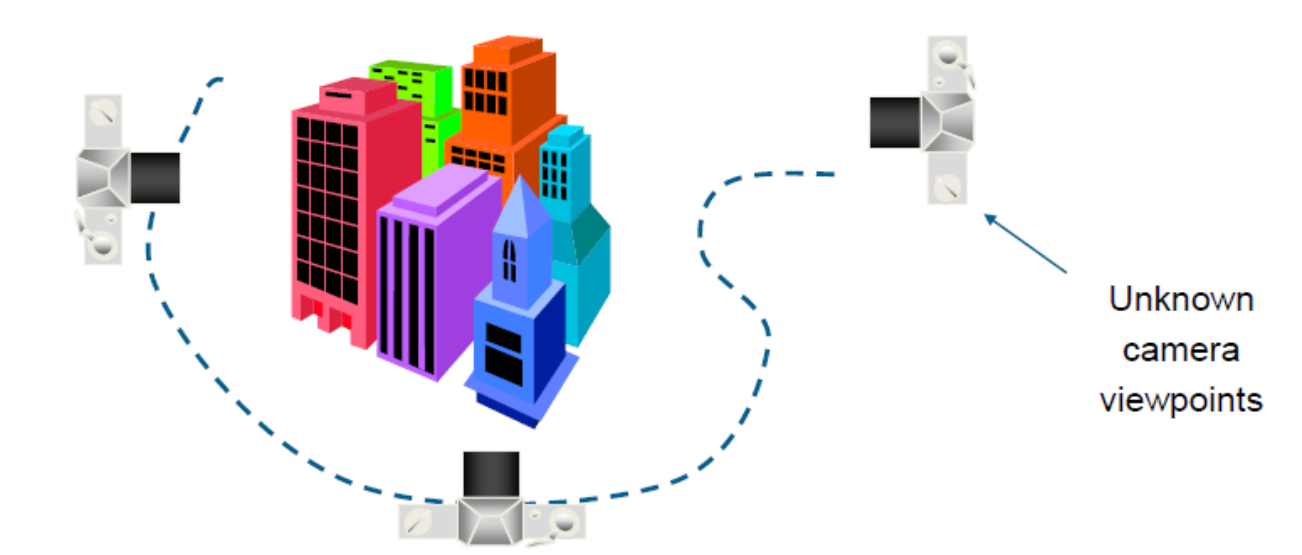
Wanted:

- Map of features
  $\mathbf{m} = \{\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_n\}$
- Path of the robot
  $\mathbf{X}_{0:k} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_k\}$



Visual SLAM

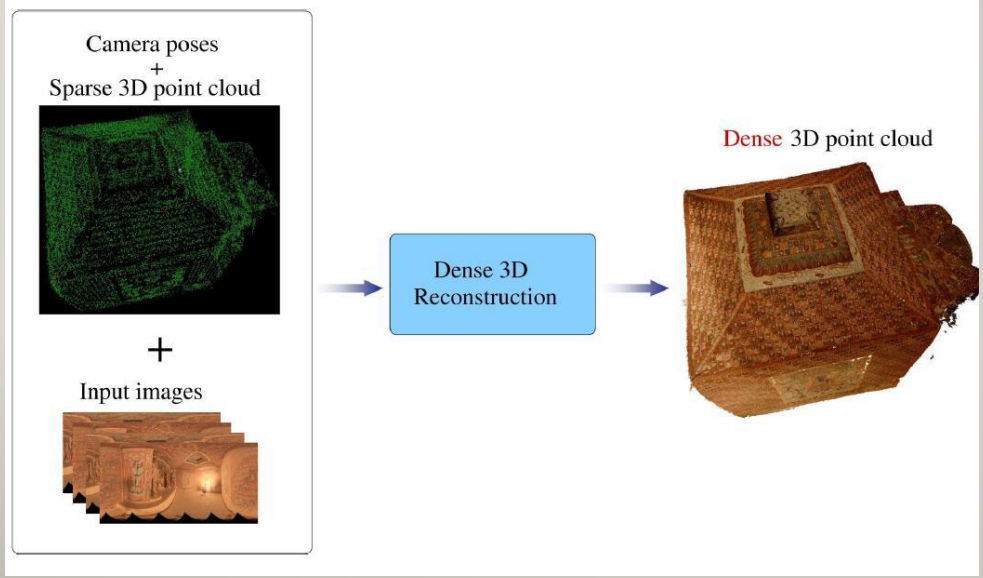
- SLAM in which camera is used as the sensor
- Largely similar to “Structure from Motion” (SfM) from vision community
- Build a map and use it at the same time – key goal in robotics and augmented reality

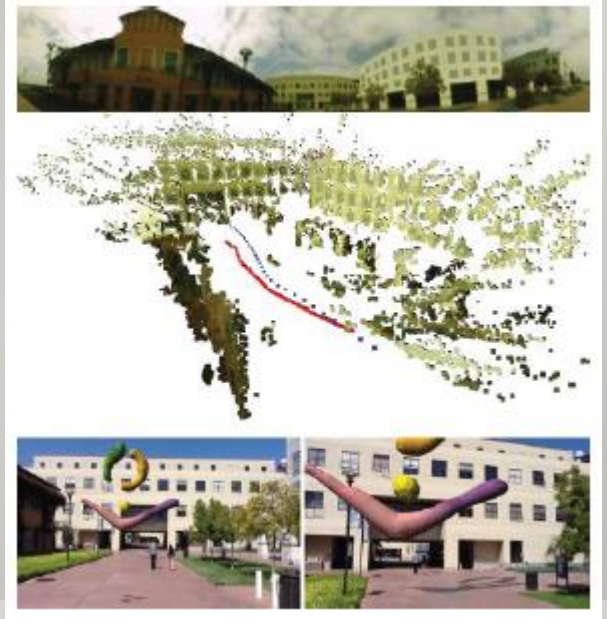


Online and Offline Structure from Motion

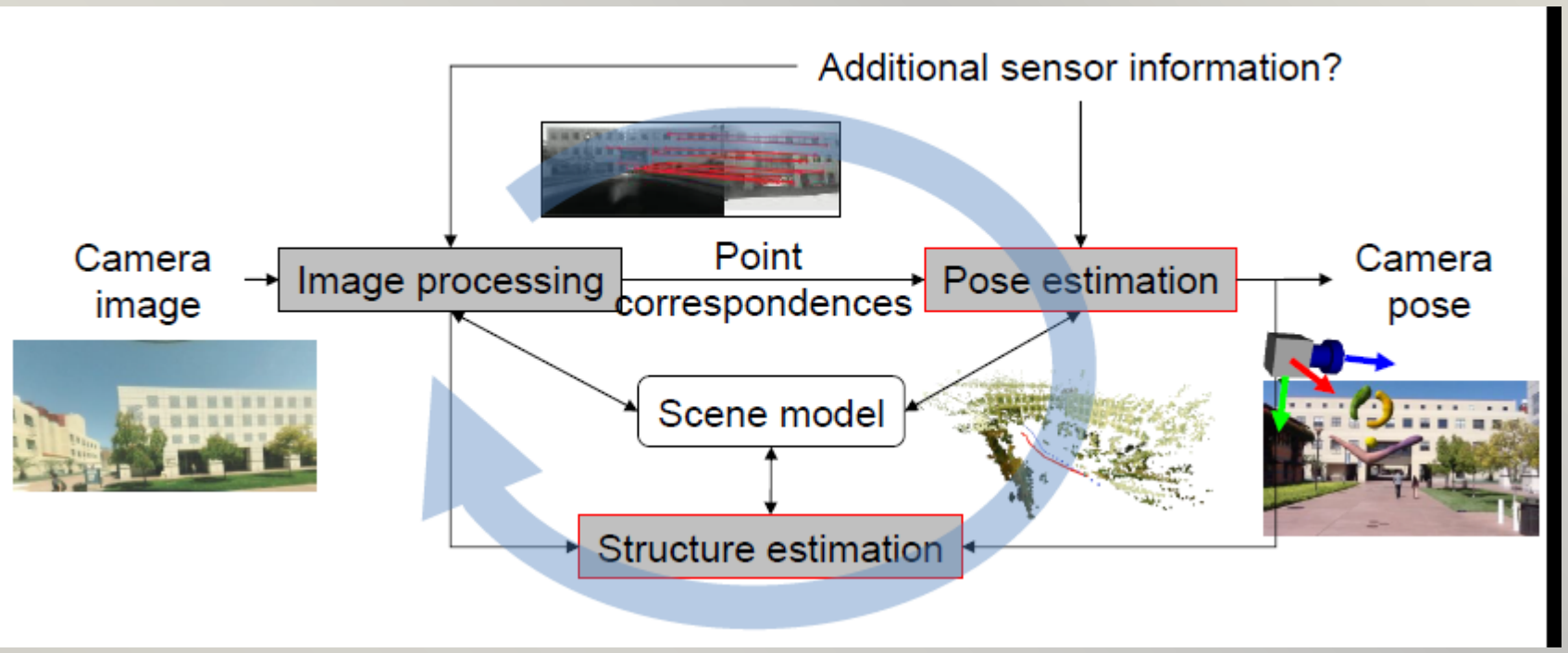
- Offline
  - E.g. as basis for dense 3D model reconstruction
  - No real-time requirements, all
  - images are available at once

- Online
  - E.g. for mobile Augmented Reality in unknown environments
  - Real-time requirements, images become available one by one, output required at each time-step



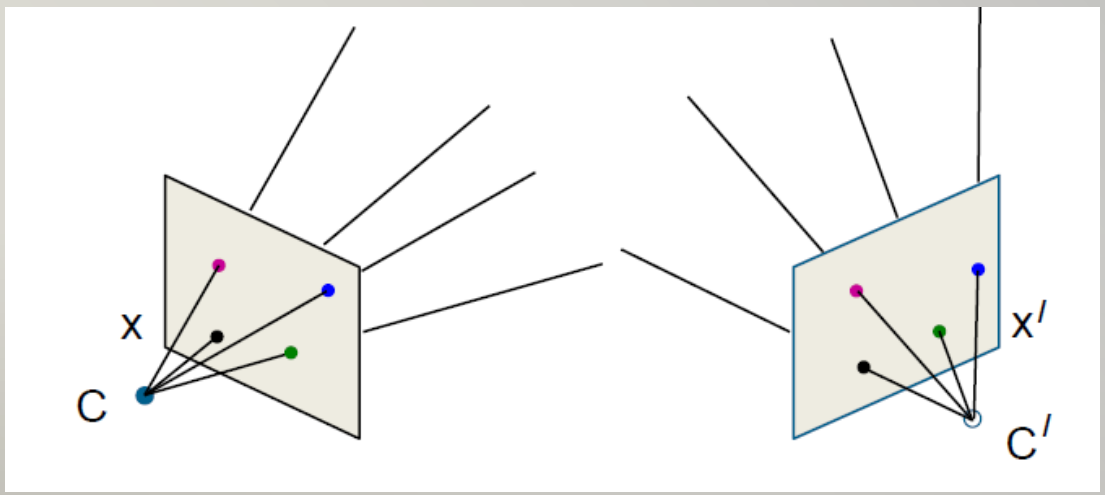


Online Structure from Motion



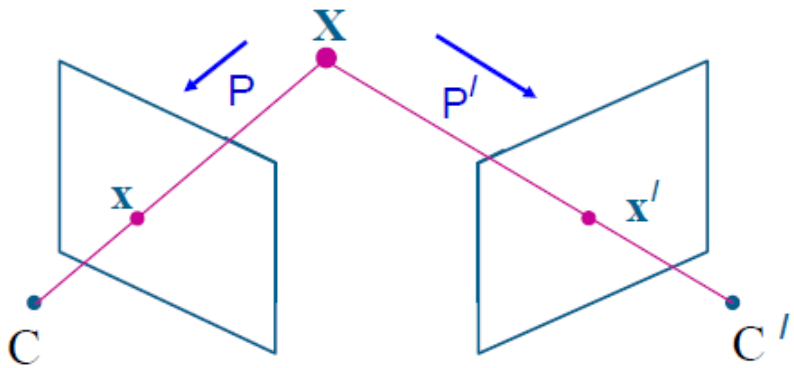
Two Views Geometry (Epipolar Geometry)

- Given image point correspondences, e.g. x_i <-> x_i',
- Determine the relative position and orientation R and t
- Involve a vision topic called Epipolar Geometry



Epipolar Geometry (In brief)

$$\mathbf{x} = \mathrm{P}\mathbf{X} \qquad \mathbf{x}' = \mathrm{P}'\mathbf{X}$$



P : 3×4 matrix

$\mathbf{X}$: 4-vector

$\mathbf{x}$: 3-vector

$$\mathrm{P} = \mathrm{K}\,[\,\mathrm{R} \mid \mathbf{t}\,]$$

internal calibration

rotation

translation

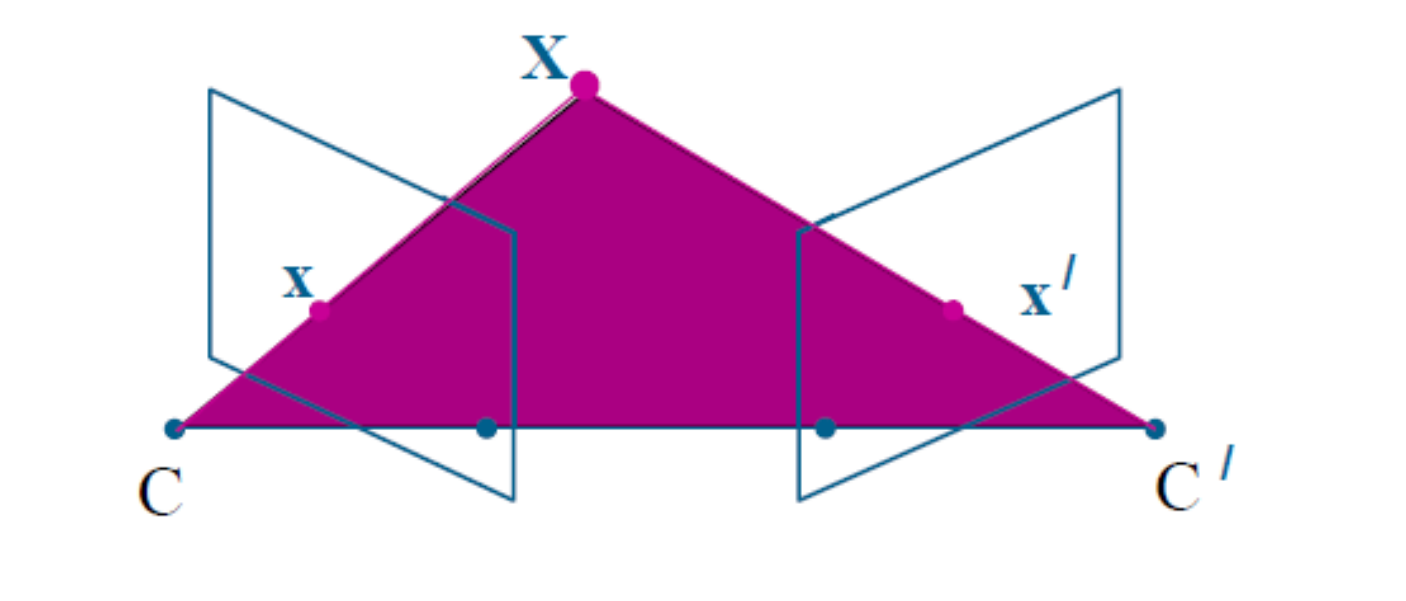
from world to camera coordinate frame

Epipolar Geometry (In brief)

first camera $P = K[I \mid 0]$

world coordinate frame aligned with first camera

second camera $P' = K'[R \mid t]$



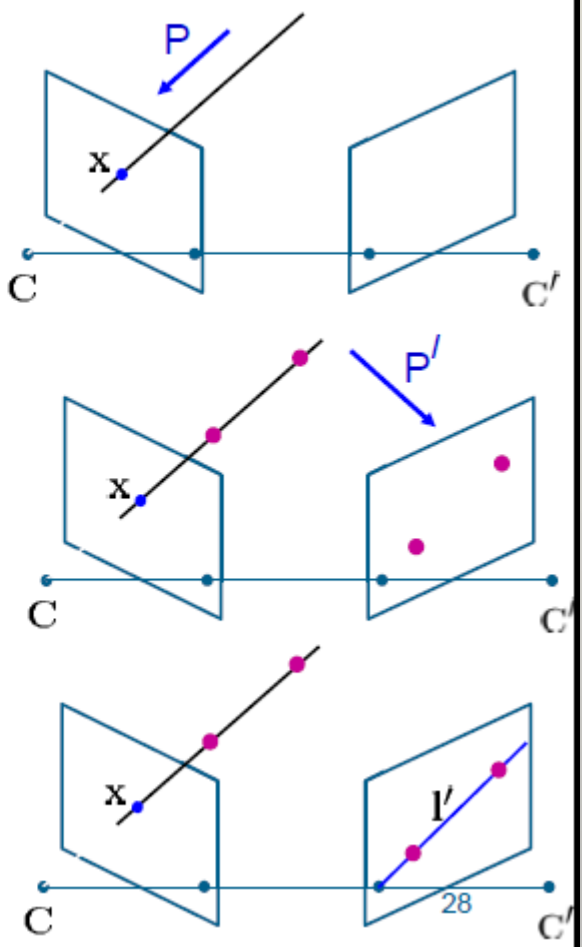
Epipolar Geometry (In brief)

<u>Outline</u>

Step 1: for a point x in the first image back project a ray with camera P

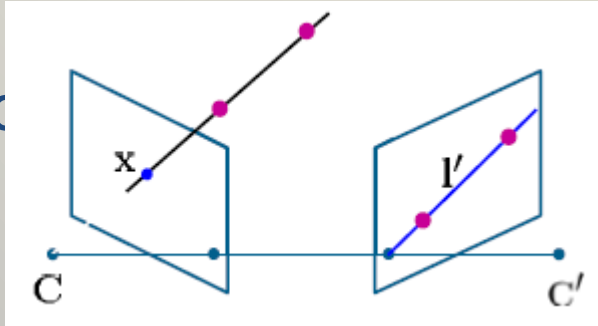
Step 2: choose two points on the ray and project into the second image with camera P'

Step 3: compute the line through the two image points using the relation $\mathbf{l}' = \mathbf{p} \times \mathbf{q}$



Epipolar Geometry (In brief)

- Finally we should get the line l' in follo



$$\mathbf{l}' = \mathbf{F}\mathbf{x} \qquad \mathbf{F} = \mathrm{K}'^{-\top}[\mathbf{t}]_\times \mathrm{R}\mathrm{K}^{-1}$$

Points $\mathbf{x}$ and $\mathbf{x}'$ correspond ($\mathbf{x} \leftrightarrow \mathbf{x}'$) then $\mathbf{x}'^\top \mathbf{l}' = 0$

- Thus:

$$\mathbf{x}'^\top \mathbf{F}\mathbf{x} = 0$$

- F here is the Fundamental Matrix

Fundamental Matrix

F is the unique 3x3 rank 2 matrix that satisfies

$$x'^{T}Fx=0 \text{ for all } x \leftrightarrow x'$$

- If we can solve F, as we should know
- K and K’ -> Using Camera Calibration
- Relative rotation and translation (R and t) can be found !!!

Solving for F (8 points algorithm)

Problem statement

Given: *n* **corresponding points** $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i, i = 1, \ldots, n\}$

compute the fundamental matrix F such that

$$\mathbf{x}'^{\top}_i \mathbf{F} \mathbf{x}_i = 0 \qquad 1 \leq i \leq n$$

Solution

Each point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ generates one constraint on $\mathbf{F}$

$$(x'_i \; y'_i \; 1) \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = 0$$

which may be written

$$x'xf_1 + x'yf_2 + x'f_3 + y'xf_4 + y'yf_5 + y'f_6 + xf_7 + yf_8 + f_9 = 0$$

$$(x'x,\ x'y,\ x',\ y'x,\ y'y,\ y',\ x,\ y,\ 1)\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{pmatrix} = 0$$

For n points

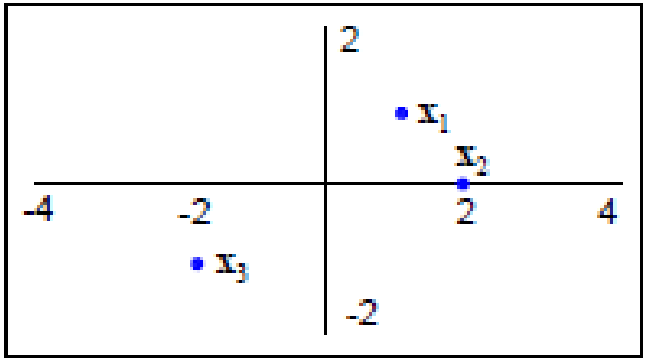
$$\mathbf{Af} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix}\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{pmatrix} = 0$$

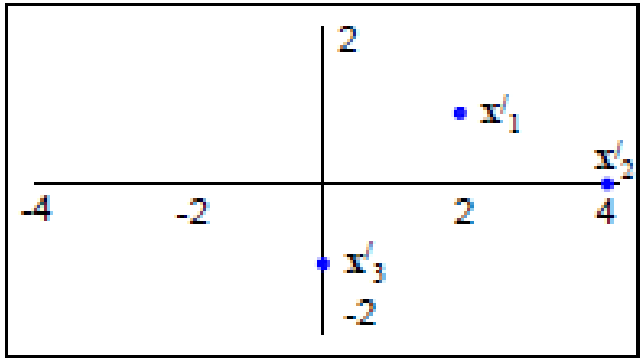
A is an n x 9 measurement matrix, and $\mathbf{f}$ is the fundamental matrix written as a 9-vector

- For 8 points, A is an 8 x 9 matrix and **f** can be computed as the null-vector of A, i.e. **f** is determined up to scale

Example: compute F from 8 point correspondences

Images from a parallel camera stereo rig – epipolar lines $y = y'$





just consider first three points

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \quad \begin{pmatrix} x'_1 \\ y'_1 \end{pmatrix} \qquad \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \quad \begin{pmatrix} x'_2 \\ y'_2 \end{pmatrix} \qquad \begin{pmatrix} x_3 \\ y_3 \end{pmatrix} \quad \begin{pmatrix} x'_3 \\ y'_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} \leftrightarrow \begin{pmatrix} 2 \\ 1 \end{pmatrix} \qquad \begin{pmatrix} 2 \\ 0 \end{pmatrix} \leftrightarrow \begin{pmatrix} 4 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} -2 \\ -1 \end{pmatrix} \leftrightarrow \begin{pmatrix} 0 \\ -1 \end{pmatrix}$$

$$\mathtt{A} = \begin{bmatrix} x'_i x_i & x'_i y_i & x'_i & y'_i x_i & y'_i y_i & y'_i & x_i & y_i & 1 \\ 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\ 8 & 0 & 4 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 2 & 1 & 0 & -2 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$\mathtt{A} = \begin{bmatrix} x'_i x_i & x'_i y_i & x'_i & y'_i x_i & y'_i y_i & y'_i & x_i & y_i & 1 \\ 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\ 8 & 0 & 4 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 2 & 1 & 0 & -2 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$\mathbf{f} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \end{pmatrix}^\top$$

satisfies $\mathtt{A}\mathbf{f} = \mathbf{0}$

write $\mathbf{f}$ in matrix form as

$$\mathrm{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

Solve “8” point problem in least square

- If we have more corresponding points , we can solve by minimizating the error linearly (i.e. least square)

Given n corresponding points (n is typically hundreds) with noise on their measured positions

For $n > 8$ point correspondences, $\mathtt{A}$ is a $n \times 9$ matrix,

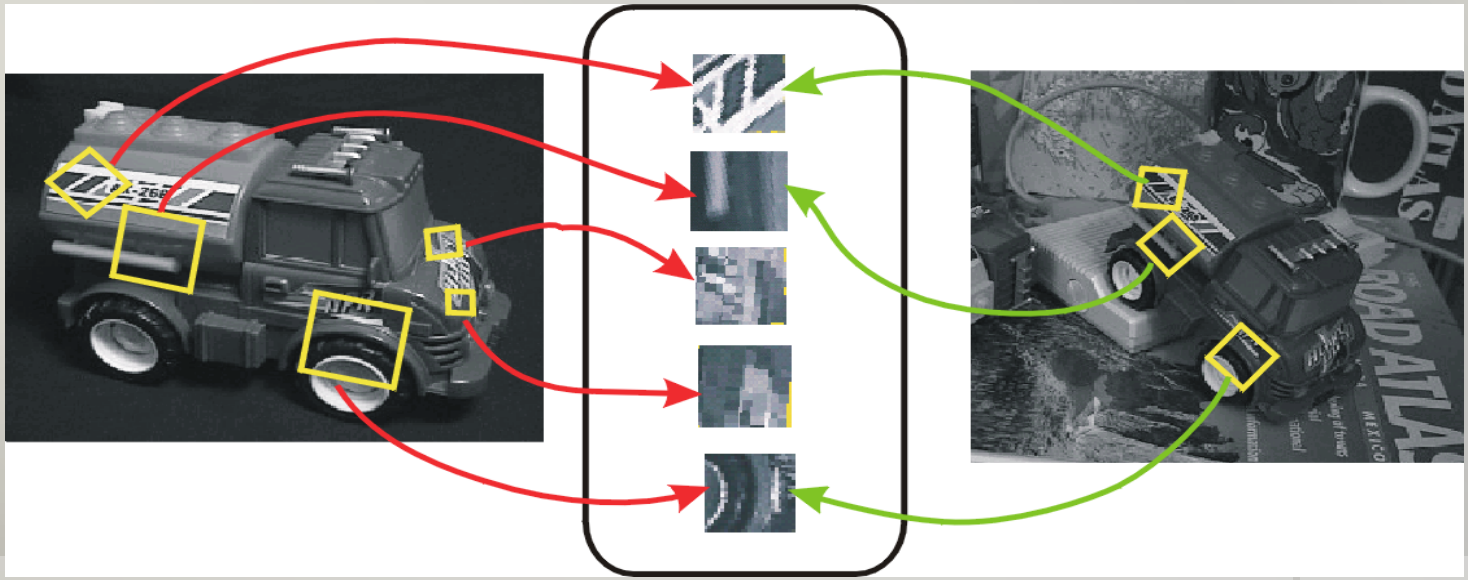
$$\begin{bmatrix} x'_1 x_1 & x'_1 y_1 & x'_1 & y'_1 x_1 & y'_1 y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_n x_n & x'_n y_n & x'_n & y'_n x_n & y'_n y_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = \mathbf{0}$$

and in general there will not be an exact solution to $\mathtt{A}\mathbf{f} = \mathbf{0}$.

A (linear) solution which minimises $||\mathtt{A}\mathbf{f}||$, subject to $||\mathbf{f}|| = 1$ is obtained from the eigenvector with least eigenvalue of $\mathtt{A}^\top \mathtt{A}$.

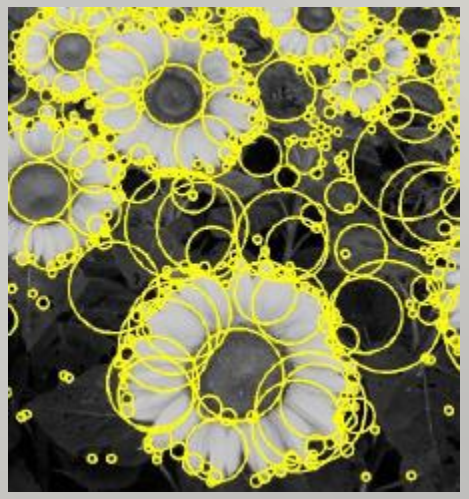
SIFT

- To complete our story, we still need method finding corresponding points
- Finding feature points is a fundamental problem in vision
- One of the most widely employed methpd : Scale Invariant Feature Transform (SIFT)
- Initially proposed for correspondence matching
  - Proven to be the most effective in such cases according to a recent performance study by Mikolajczyk & Schmid (ICCV ’03)



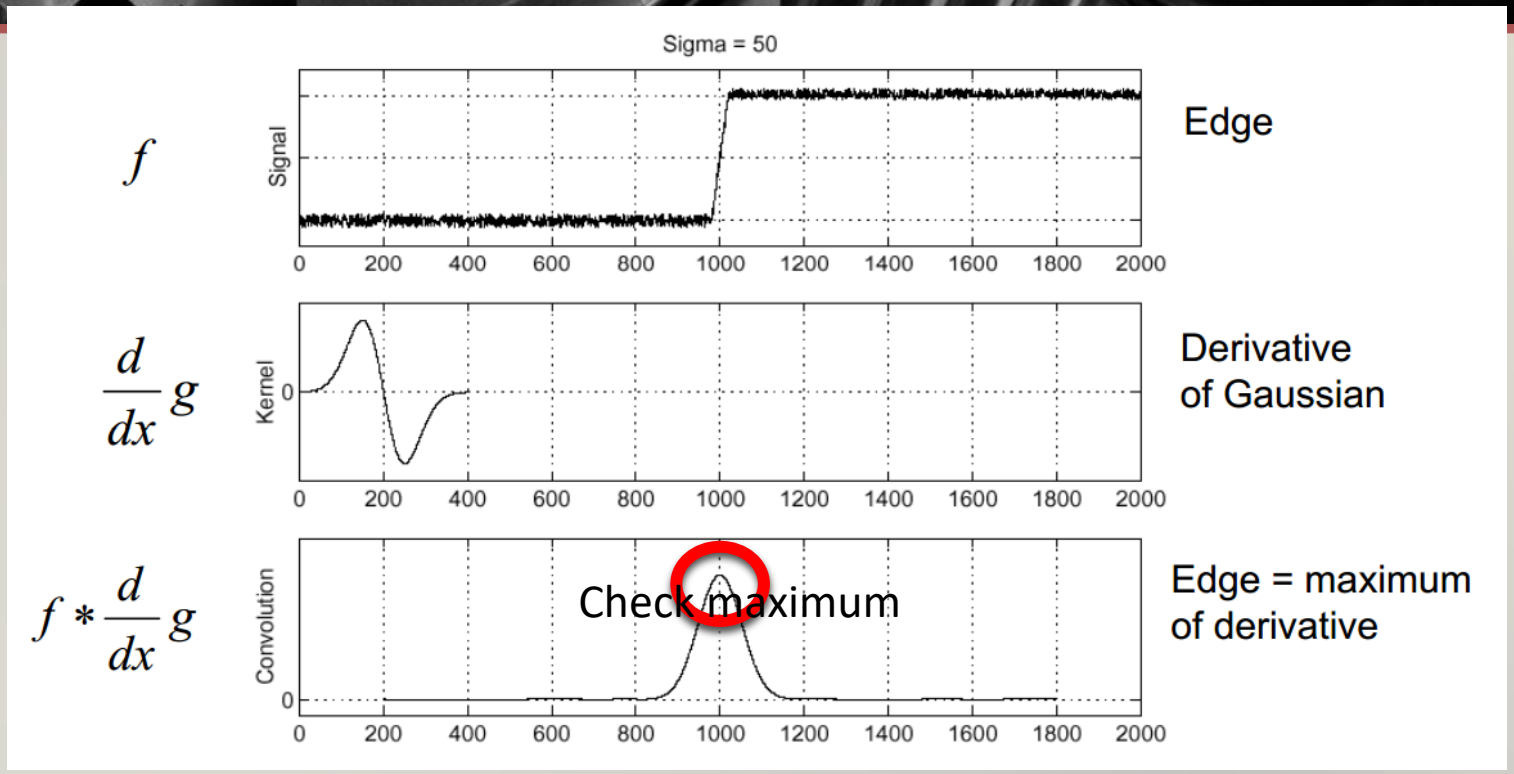
Blob Detection

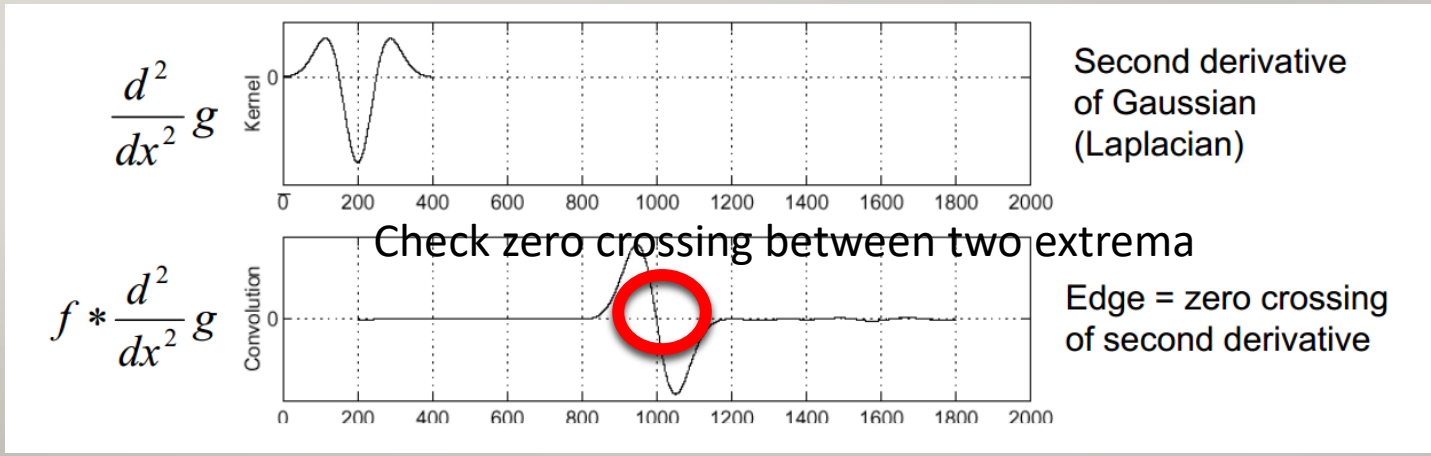
- Feature are common referred as high frequency components in data/content
- But to detect feature region, our interest is to find its center and size
- Instead of extracting edges or corner
- We are going to detect "**blob**"



Blob Detection

- Classic edge detection or extraction used to compute the first derivative of the intensity change
- Then, detect the maximum of it
- Difference of Gaussians (DoG) is used to extract Blob
  - It is like Laplacian of Gaussian (2nd derivative of Gaussian)
  - Although it is possible to detect edge or corner too



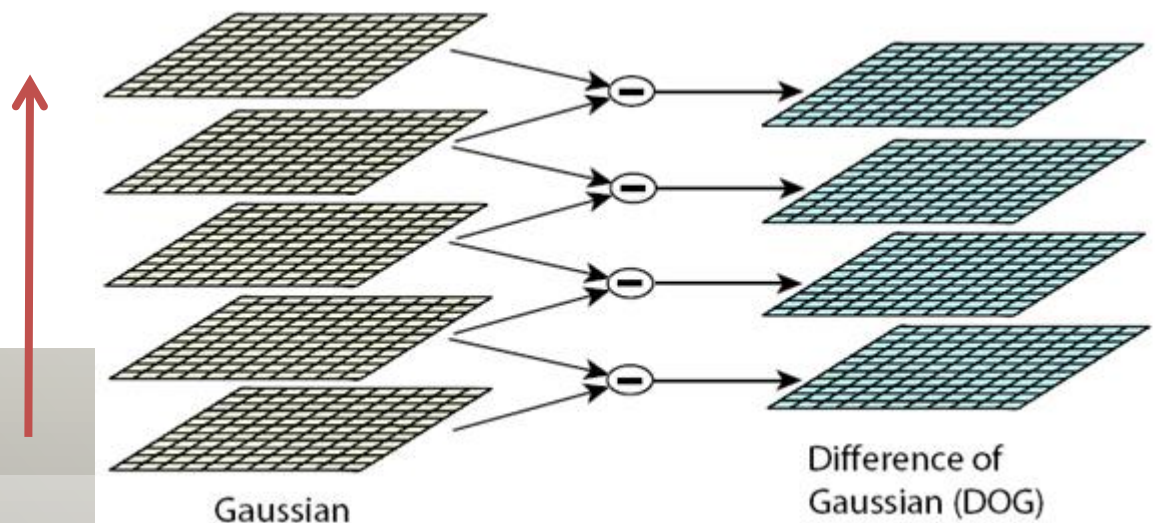


Source: S. Seitz

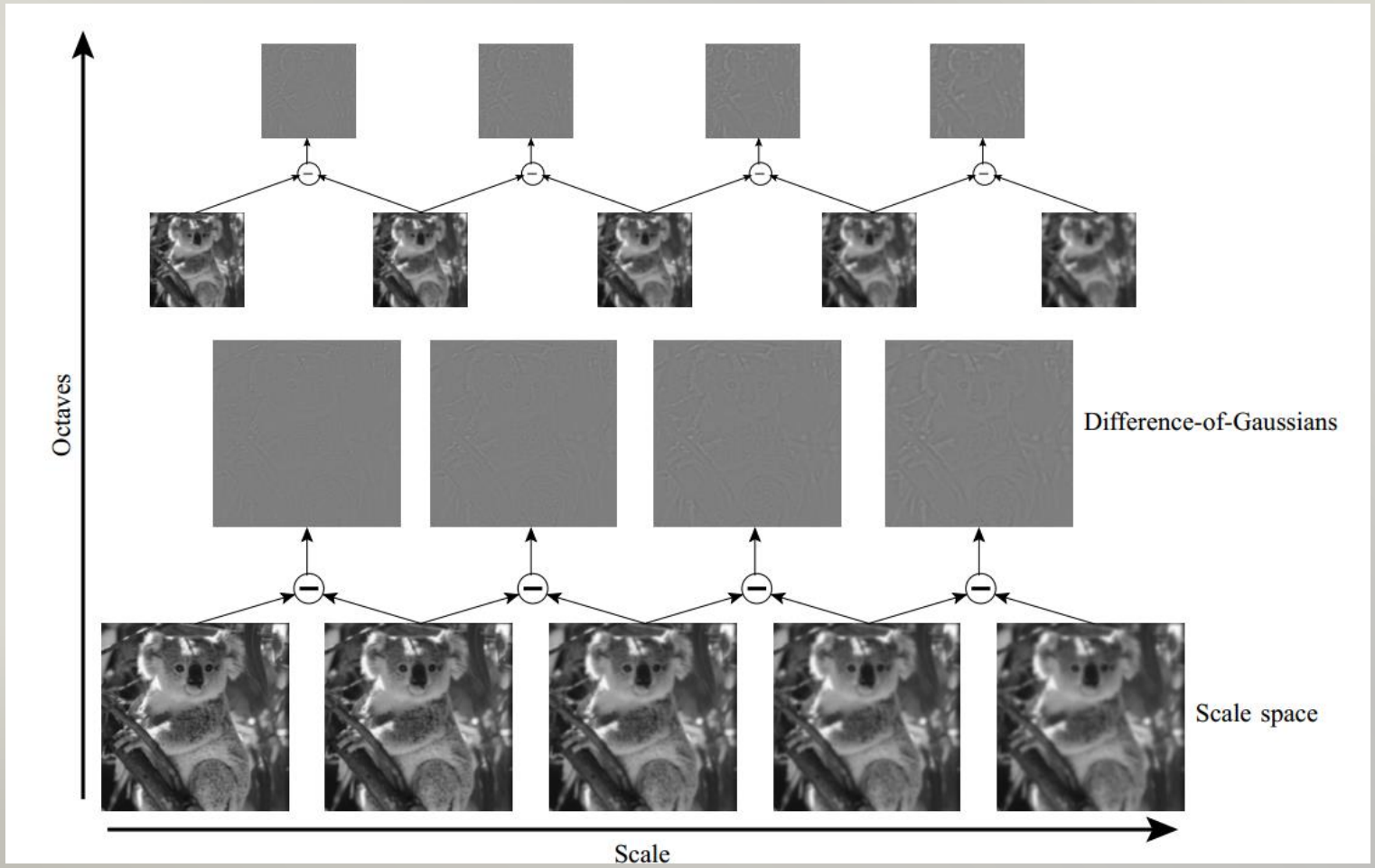
Lowe's Pyramid Scheme

- However, the Pyramid is not as simple as we used in hierarchical search in MPEG
  - In each of the level, the image is repeatedly convolved with Gaussian (increasing σ) to produce a set of scale space images
- Adjacent Gaussian are subtracted to produce DoG (Difference of Gaussian)

Increasing σ



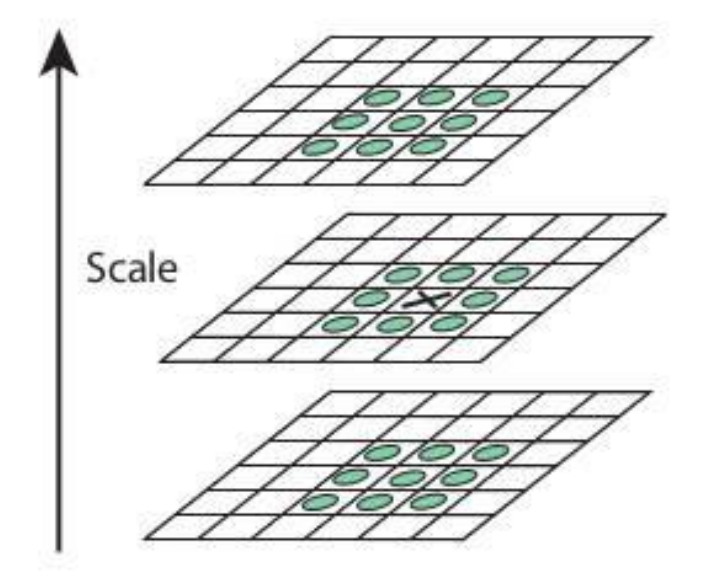
Lowe's Pyramid Scheme



From J. Almeida

Scale-space Extrema Detection

- Each pixel in the DoG of the image compared with
  - its 8 neighbours in same scale
  - 9 pixels in next scale
  - 9 pixels in previous scales.
- If it is a local extrema, it is a potential keypoint.
- It basically means that keypoint is best represented in that scale



Keypoint Filtering and Localization

- DoG has higher response for edges, so edges also need to be removed
- Eliminating edges
  - Similar to Harris corner detector
  - using Trace and Determinant of Hessian

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r}$$

Orientation Assignment

- The orientation to each keypoint is used as the descriptor

  invariance to image rotation

- The orientation can be measured by the intensity change across pixels, i.e. Intensity gradient

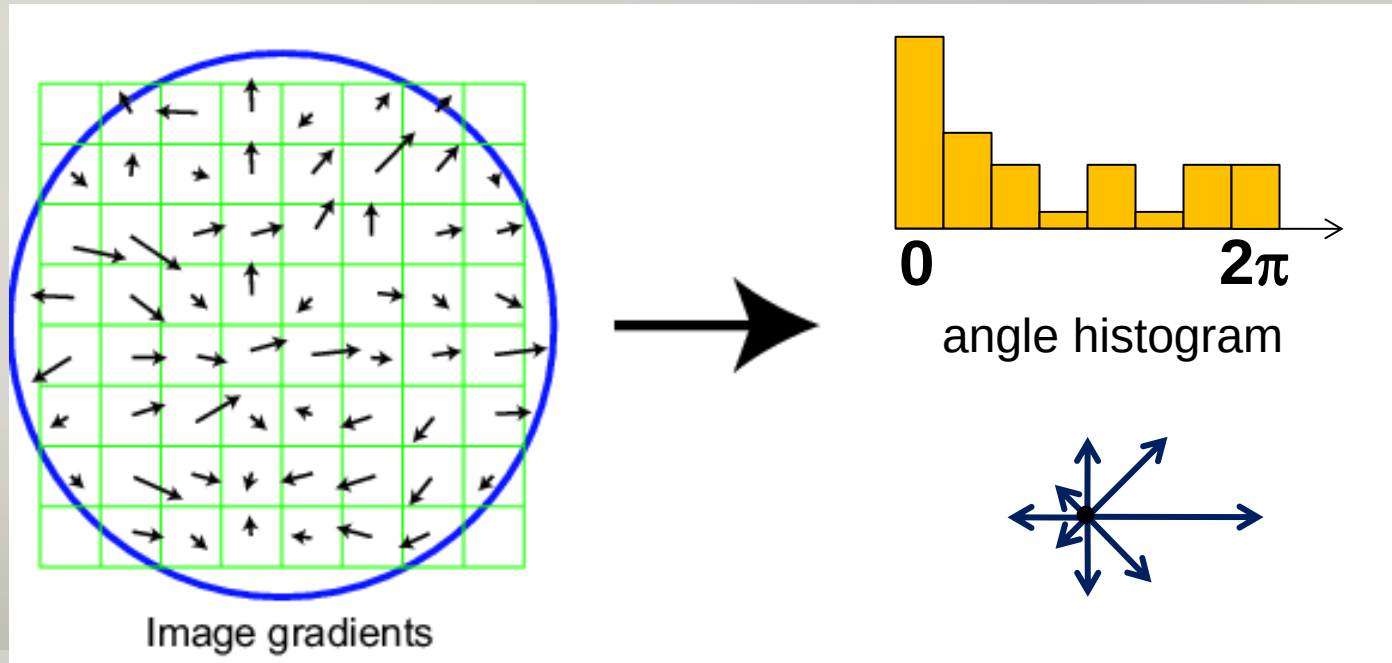
  m: magnitude , theta: angle

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

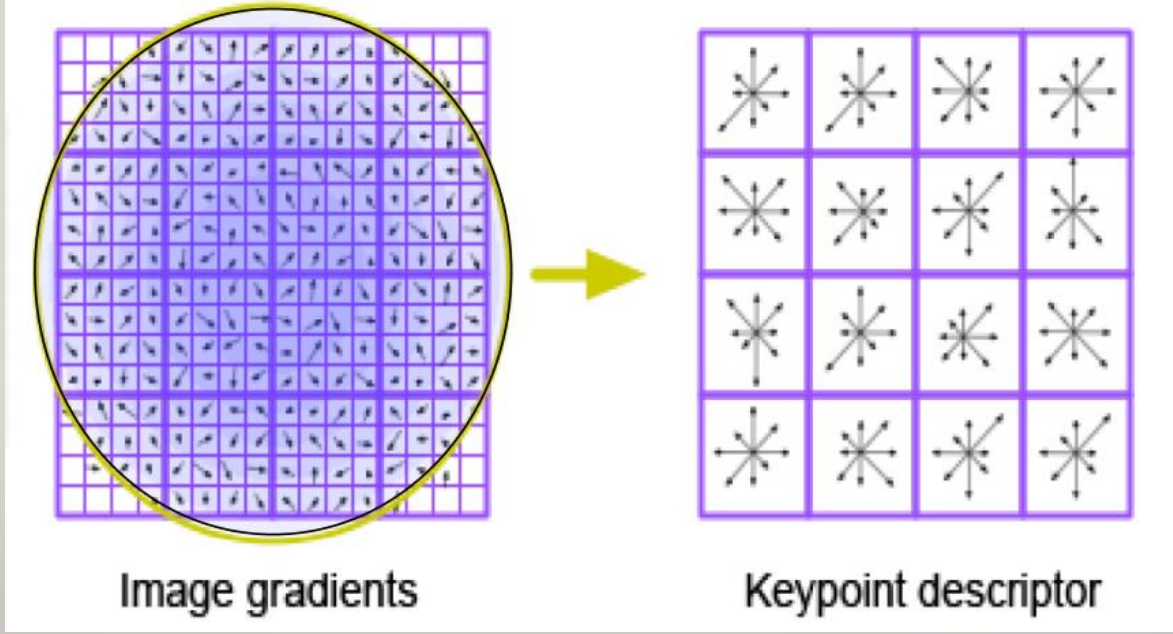
Orientation Assignment

- However, under the same region, different pixel can have different image gradients
- We therefore, collect them to form a histogram of angles (each bin for an angle range)



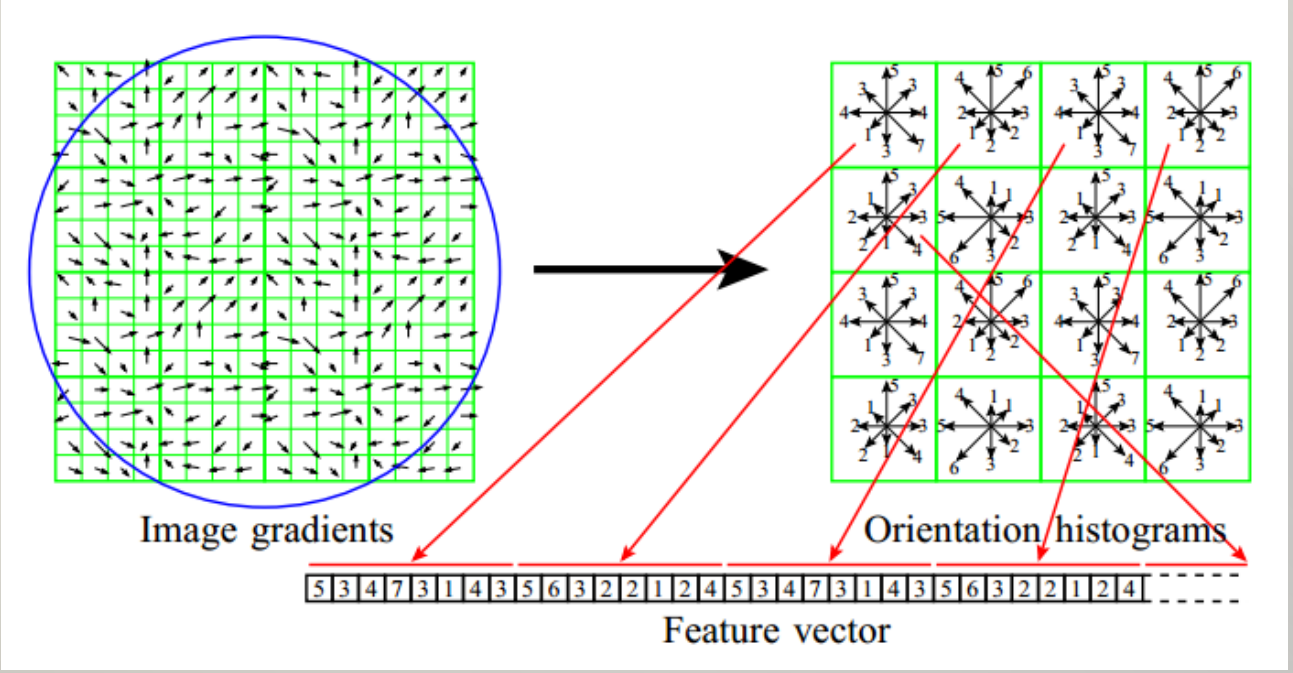
Orientation Assignment

- Takes 16x16 square window around feature
- Then, divide it into 4*4 subregions
- For a 8 bins (8 directions) histogram for each subregion



Feature Vector Composition

- Finally form the feature descriptor with 8x4x4 = 128 dimensions

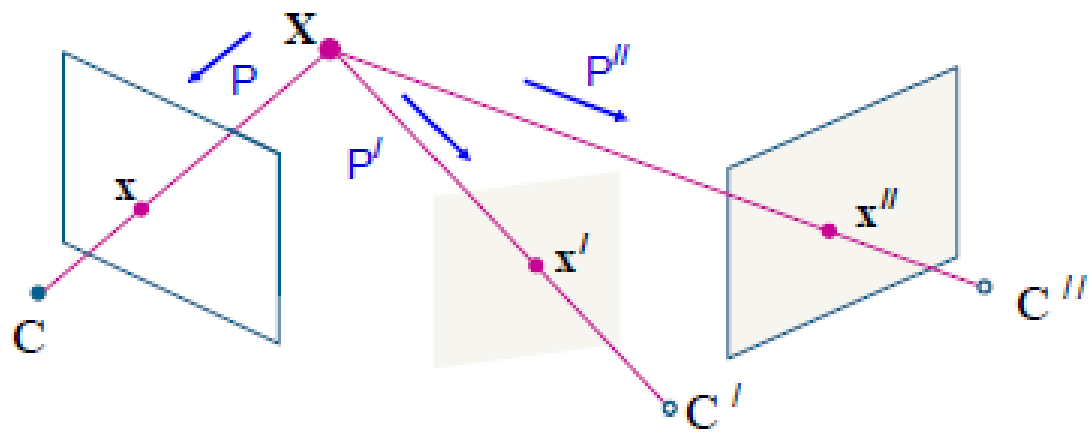


Multiview Geometry

- A moving camera should obtain more than 2 views in its motion
- Multiview geometry try to take into account more view's correspondence

For 3 views the cameras are P, P′ and P″, and a 3D point is imaged as

$$\mathbf{x} = P\mathbf{X} \quad \mathbf{x}' = P'\mathbf{X} \quad \mathbf{x}'' = P''\mathbf{X}$$



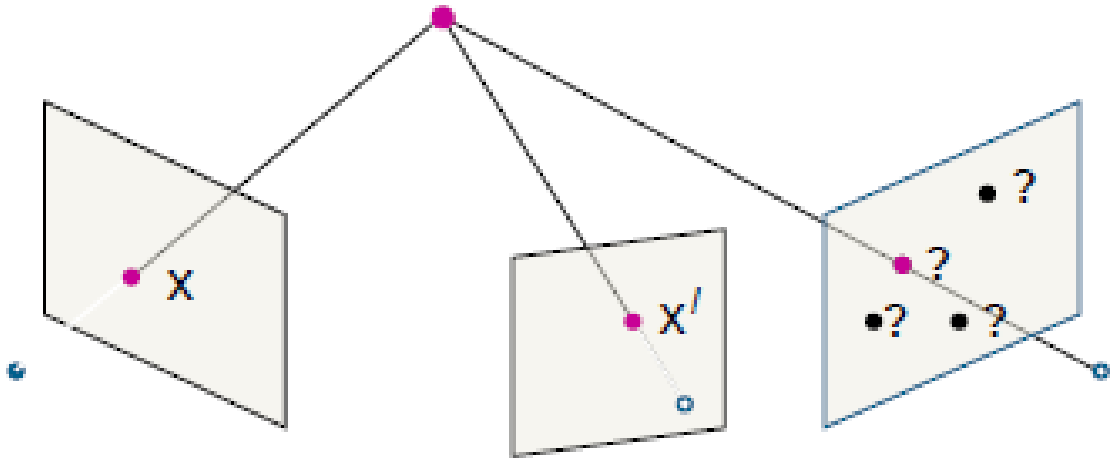
For m views, a point $\mathbf{X}_j$ is imaged in the "i" th view as

$$\mathbf{x}^i_j = P^i \mathbf{X}_j$$

Multiview Geometry

Given: the cameras P, P' and P'', and matching points $\mathbf{x}$ and $\mathbf{x}'$

Find: the matching point in the third view



Algorithm:

- compute the 3D point from $\mathbf{x}$ and $\mathbf{x}'$ and project it into the third view
- the matching point coincides with the projected point

Point constraint is stronger than epipolar constraint (line)

Bundle Adjustment

Given: n matching image points $\mathbf{x}^i_j$ over m views

Find: the cameras P^i and the 3D points $\mathbf{X}_j$ such that $\mathbf{x}^i_j = P^i \mathbf{X}_j$

$$\min_{P^i\ \mathbf{X}_j} \sum_{j \in \text{points}} \sum_{i \in \text{views}} d\left(\mathbf{x}^i_j,\ P^i \mathbf{X}_j\right)^2$$

number of parameters

- for each camera there are 6 parameters
- for each 3D point there are 3 parameters

a total of 6 m + 3 n parameters must be estimated

- e.g. 50 frames, 1000 points: 3300 unknowns

Bundle Adjustment

- Nonlinear estimation problem: use e.g. Levenberg-Marquard, start at the linear solution Open source libraries available, e.g. Sparse Bundle Adjustment (SBA)

$$\hat{x} = \arg\min \sum_{t=1}^{k} \sum_{l=1}^{n} r^{(i)}$$

Estimate

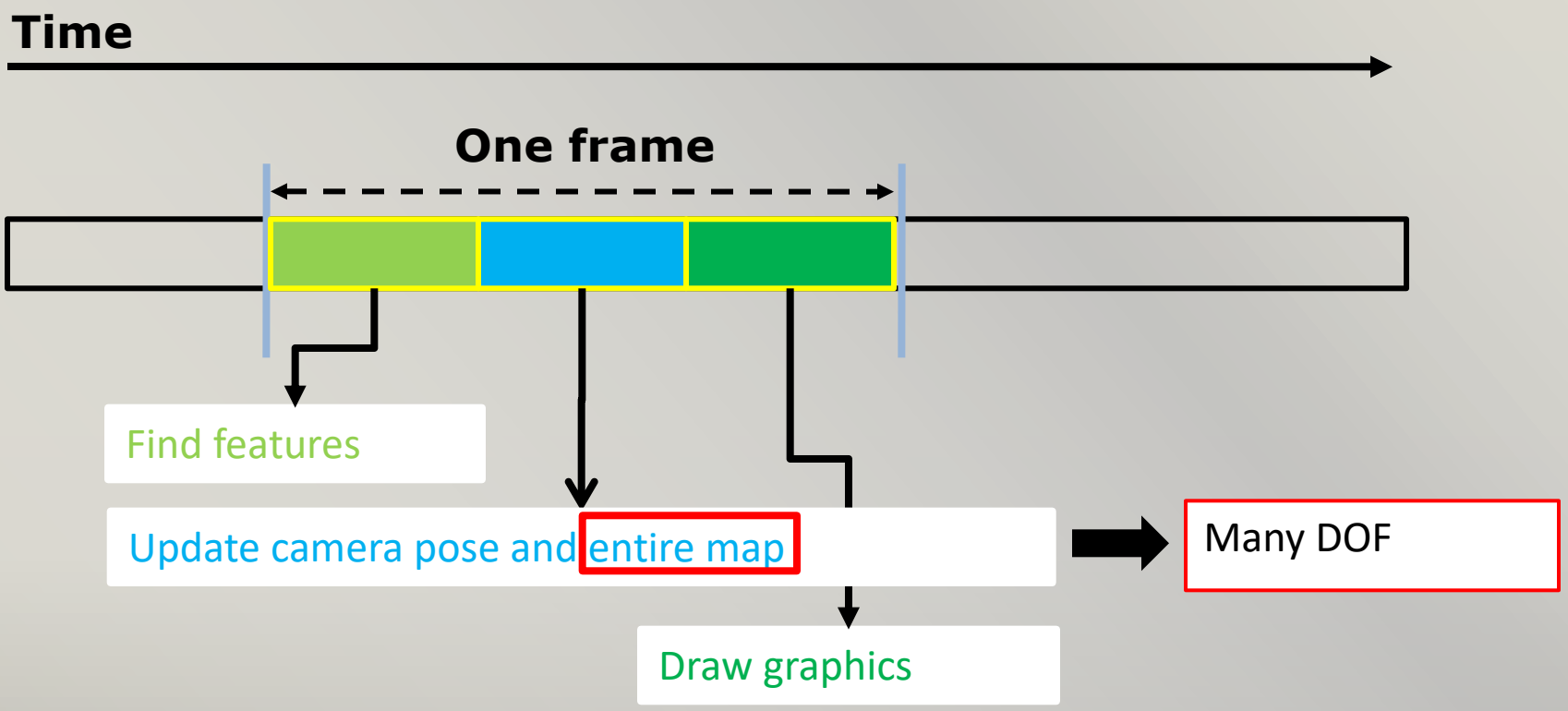
Minimize over parameter vector containing all camera poses and 3D points

Over all frames and 3D points

Residual/reprojection error

Frame by Frame SLAM

- It is a trivial approach to apply Visual SLAM in every frame



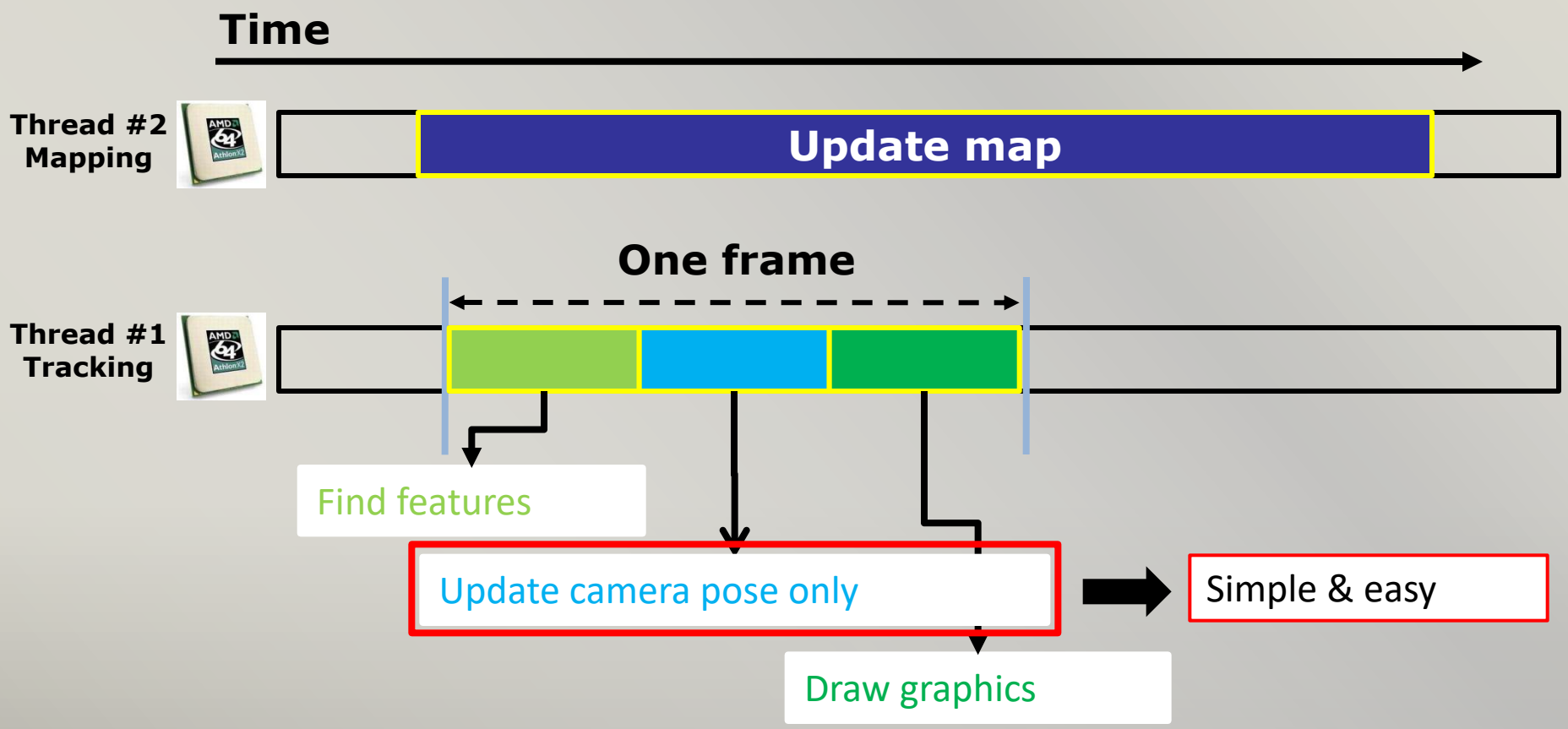
Frame by Frame SLAM

- SLAM
  - However, updating entire map every frame is so **expensive**!!!
- Proposed approach
  - Don’t update the map every frame : **Keyframes**
  - **Split the tracking and mapping into two threads**

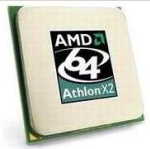
Parallel Tracking and Mapping (PTAM)

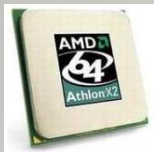
- Proposed method

  - Split the tracking and mapping into two threads



Parallel Tracking and Mapping (PTAM)





Tracking thread:

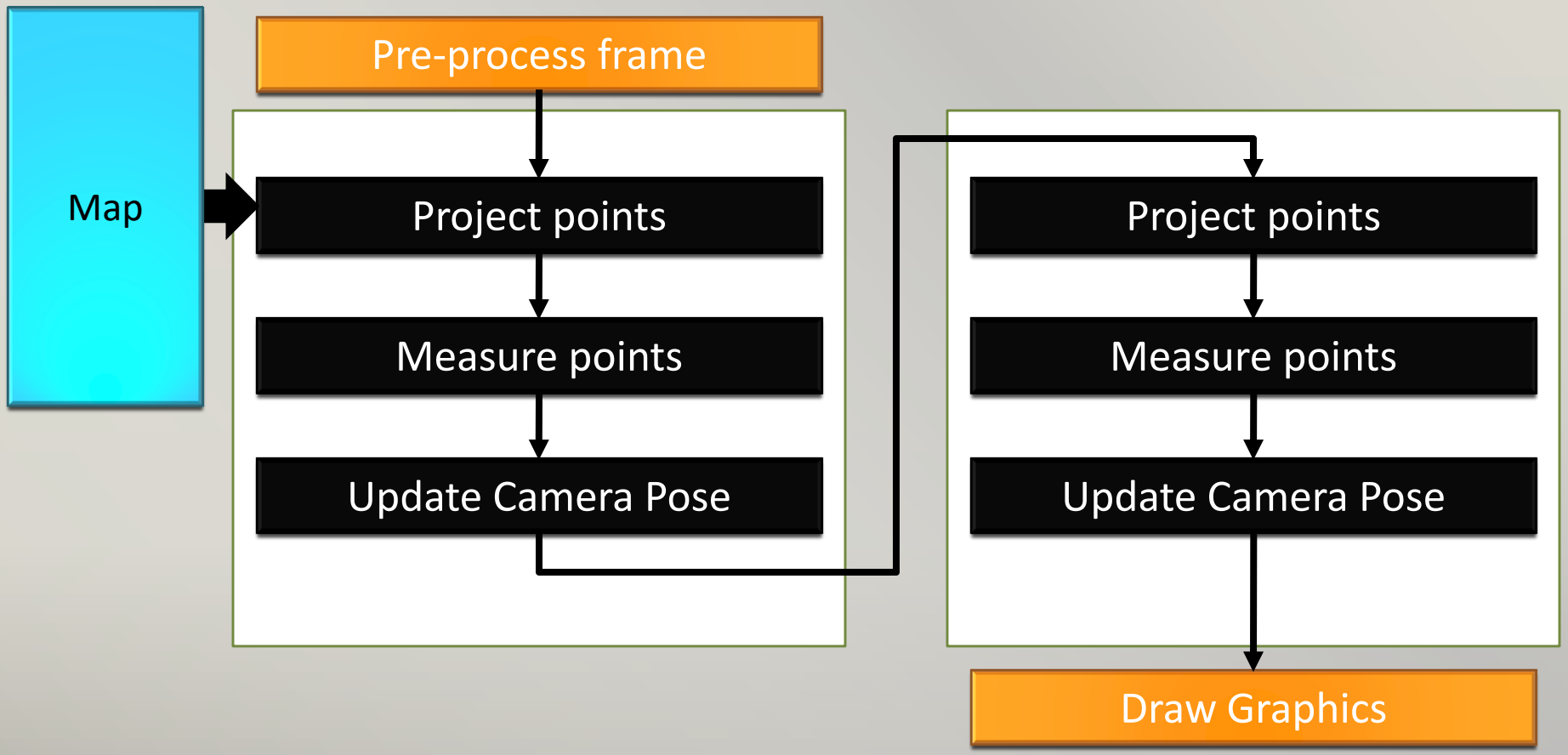
- Responsible estimation of camera pose and rendering augmented graphics
- Must run at 30 Hz
- Make as robust and accurate as possible

Mapping thread:

- Responsible for providing the map
- Can take lots of time per key frame
- Make as rich and accurate as possible

Tracking thread

- Overall flow

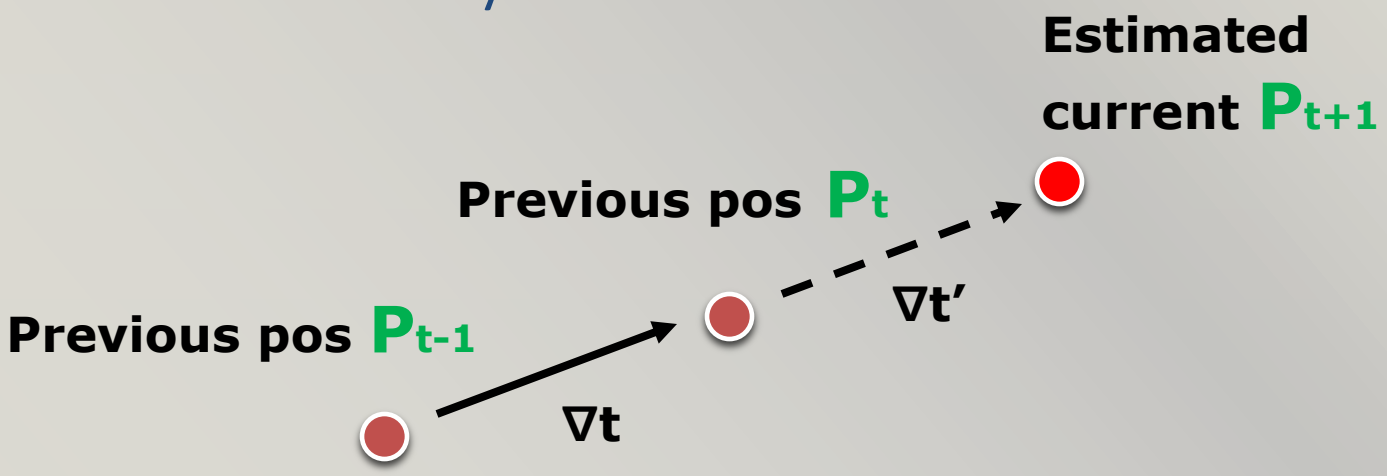


Pre-process frame

- Make for pyramid levels
- Detect Fast corners

Project Points

- Use motion model to update camera pose
  - Constant velocity model



$$V_t = (P_t - P_{t-1}) / \nabla t$$

$$P_{t+1} = P_t + \nabla t'(V_t)$$

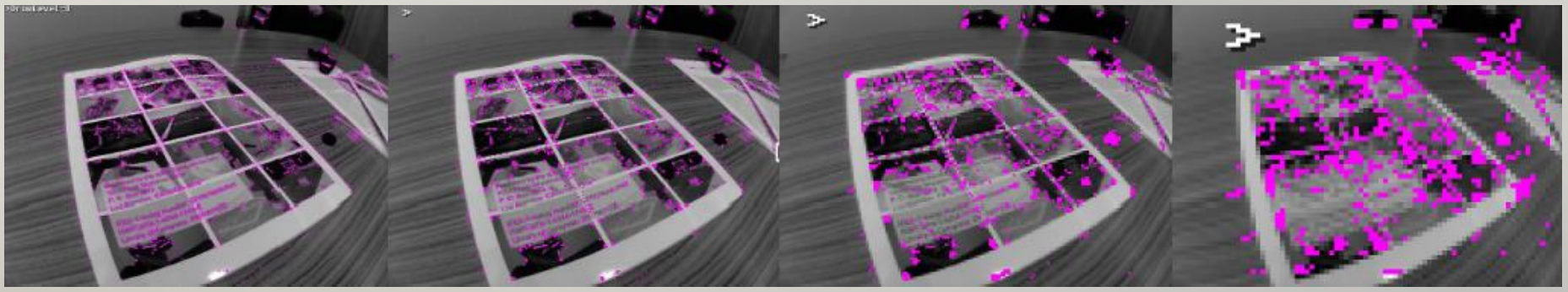
Project Points

- Choose subset to measure
  - ~ 50 biggest features for coarse stage
  - 1000 randomly selected for fine stage

1000

~50

640x480 **320x240** **160x120** **80x60**



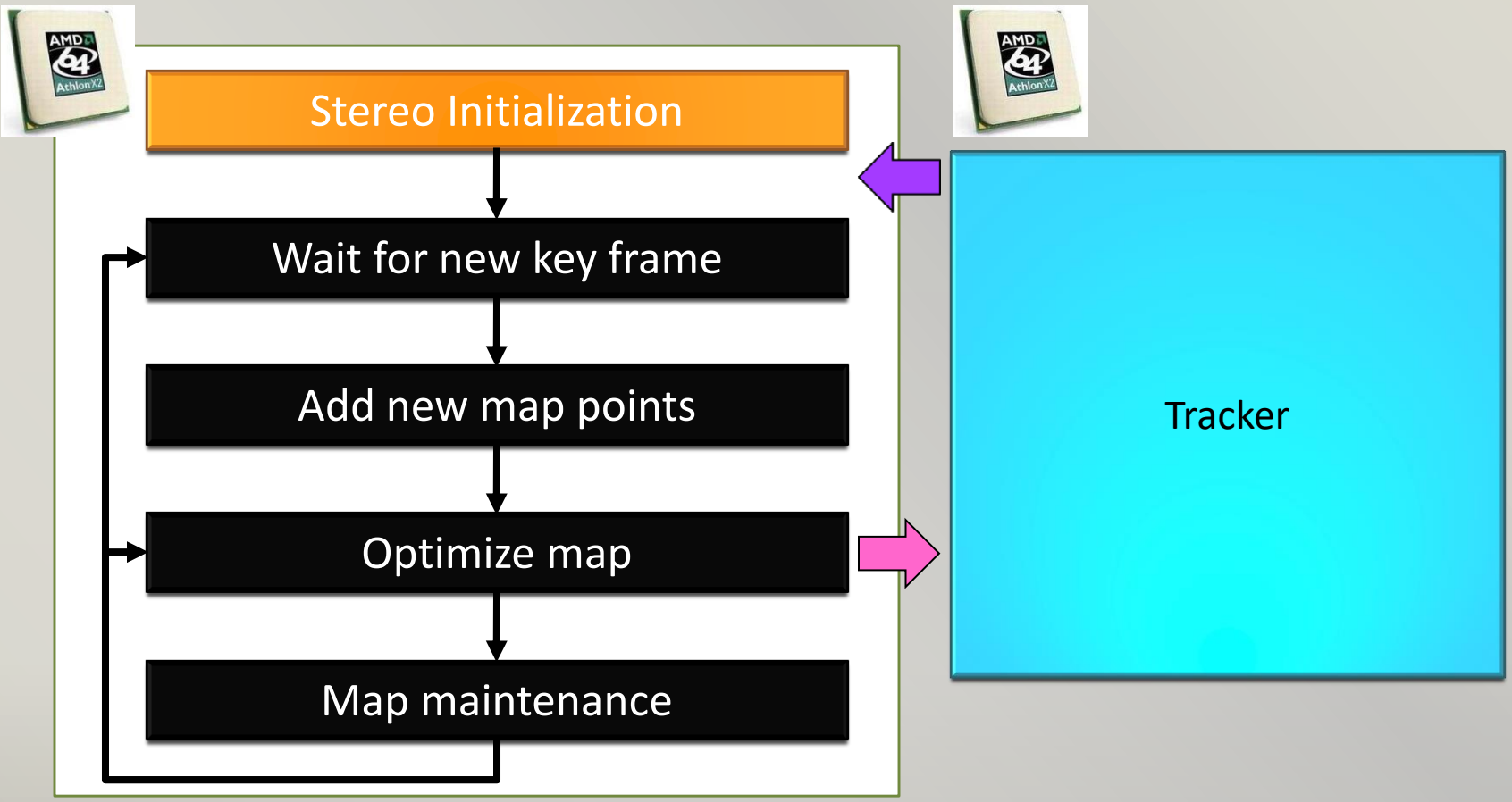
Measure Points

- Generate 8x8 matching template (warped from source key-frame:map)

- Search a fixed radius around projected position
  - Use Zero-mean SSD
  - Only search at Fast corner points

- Finally, update camera pose based on the matched project points

Mapping thread



Wait for new key frame and add map points

- Key frames are only added if :
  - There is a sufficient baseline to the other key frame
  - Tracking quality is good
  - Key frame (4 level pyramid images and its corners)
- When a key frame is added :
  - All points in the map are measured in the key frame
  - New map points are found and added to the map
  - Check all maximal FAST corners in the key frame

    Check score and if already in map

Optimize map and map maintenance

- Use batch SFM method: Bundle Adjustment
- Adjusts map point positions and key frame poses
- Minimize reprojection error of all points in all keyframes (or use only last N key frames)

- When camera is not exploring, mapping thread has idle time
- Data association in bundle adjustment is reversible
- Re-attempt outlier measurements
- Try measure new map features in all old key frames

Comparison to EKF-SLAM

- More Accurate
- More robust
- Faster tracking

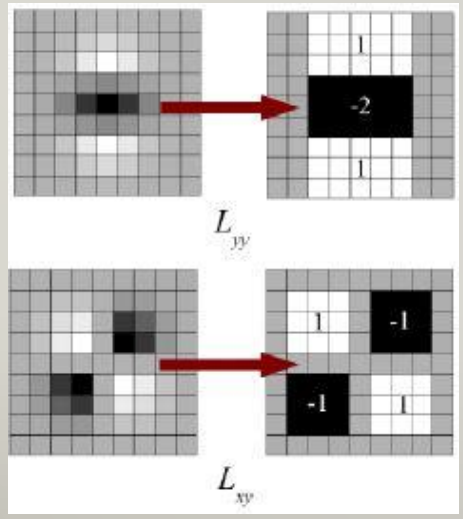
Total	19.2 ms
Key frame preparation	2.2 ms
Feature Projection	3.5 ms
Patch search	9.8 ms
Iterative pose update	3.7 ms

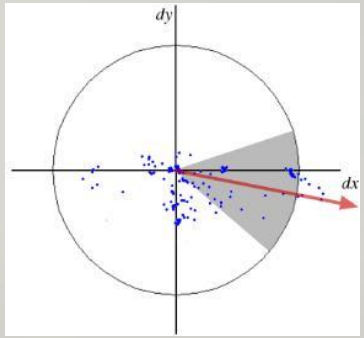
SLAM based AR

Proposed AR

Latest Advancement

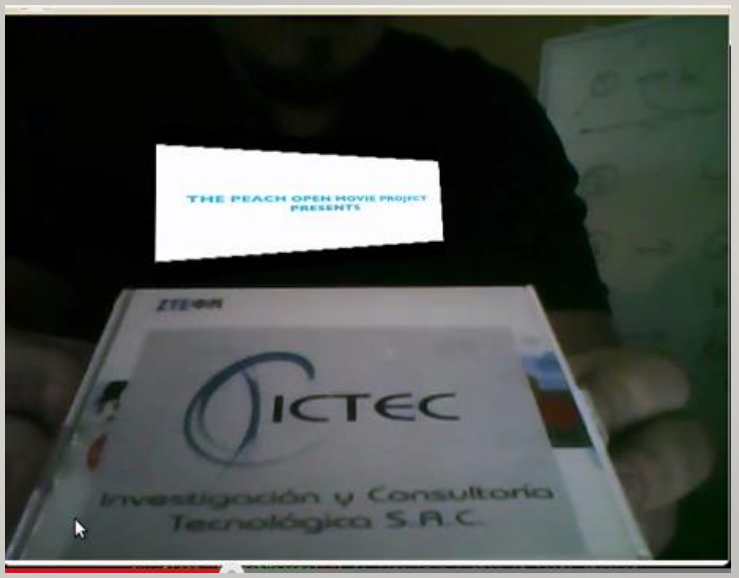
- Marker based AR
  - Arbitrary marker
- Feature matching
  - Algorithm like SURF/SIFT





Introduction to SURF (Speeded-Up Robust Features)
http://docs.opencv.org/trunk/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html

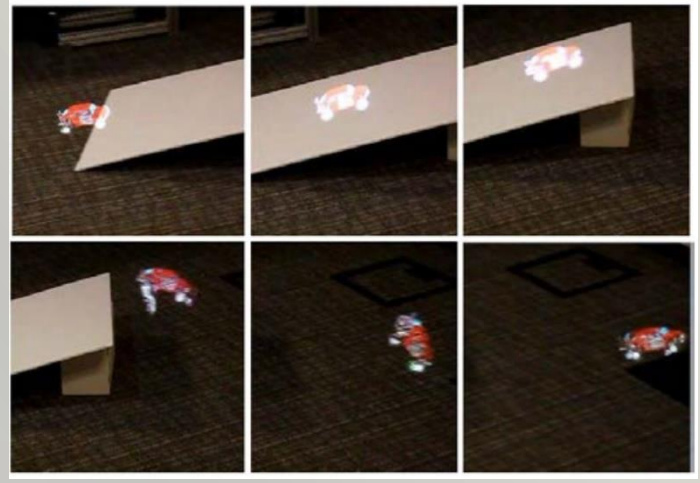
http://disneyandmore.blogspot.hk/2011/08/hong-kong-disneyland-halloween-will.html



Markerless Augmented Reality using OpenCV & SURF technique
http://www.youtube.com/watch?v=KLm9HcdtrXI

Latest Advancement (cont')

- Projector based AR
  - Geometry aware projection
  - Mobile Projector



http://www.vs.inf.ethz.ch/edu/FS2013/DS/reports/AlexanderCebulla_ProjectionBasedAugmentedReality_report.pdf

- AR by Depth sensing
  - Based on the latest advancement of depth camera
  - Extract body parts or other objects

http://www.youtube.com/watch?v=051heFG3onk

Hololens

- Mixed reality smartglasses by Microsoft
- Development Edition, shipped on the 30th of March 2016 with price of USD$3,000
- Different to Oculus Rift or HTC Vive, it can work independent to the computer
- Has its own processor and connect wirelessly to the computer

Hololens

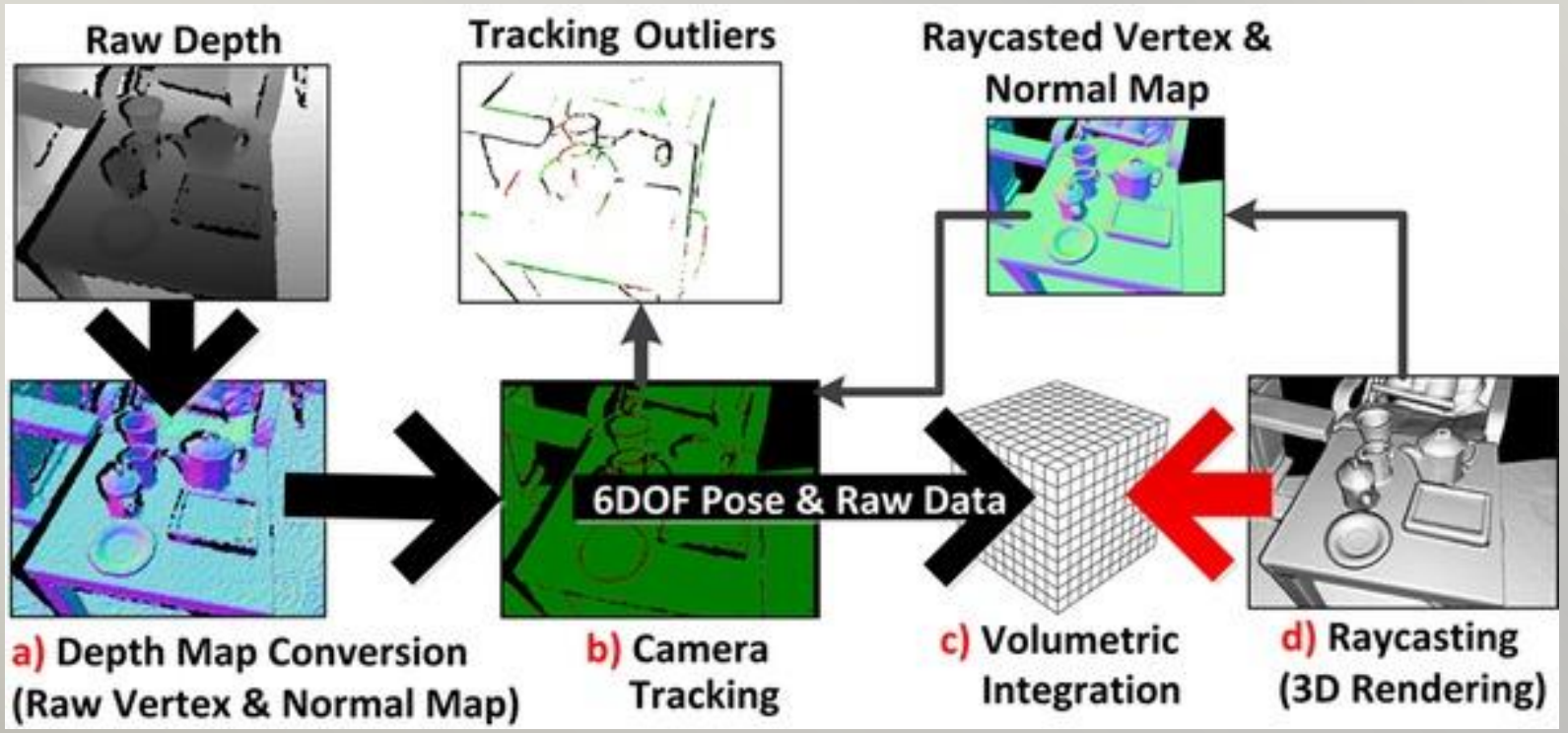
- See-through holographic lenses
- IMU and Depth sensing camera
  - miniature version of the Kinect camera
- Spatial sound unit
- HPU (holographic processing unit)
  - With CPU, GPU, RAM all on a single motherboard

Microsoft HoloLens: The Science Within - Keeping Holograms in Place

- https://youtu.be/TneGSeqVAXQ

Hololens related Technologies: Kinect Fusion

- Little information can be found on the methods of object tracking in Hololens
- Based on its minature depth sensor used, a closely related project from Microsoft called Kinect Fusion should give some hints to us
- In Kinect Fusion, it tries to reconstruct scene structure based on Kinect



Applications

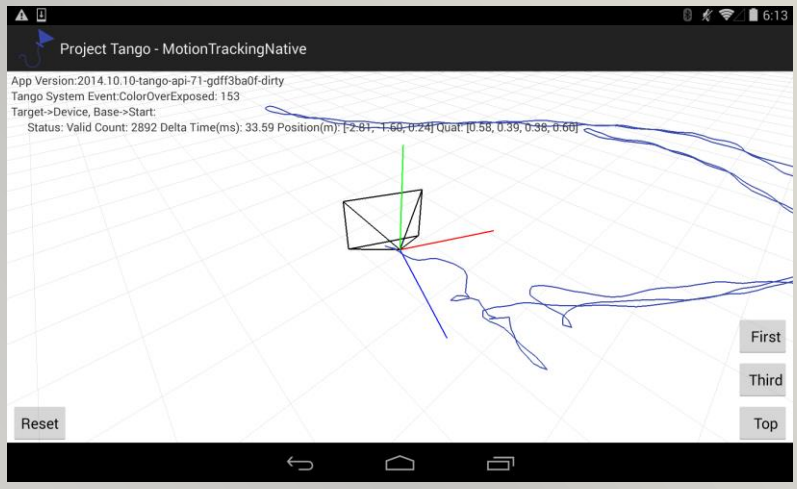
- holoportation: virtual 3D teleportation in real-time
  - https://www.youtube.com/watch?v=7d59O6cfaM0&feature=youtu.be

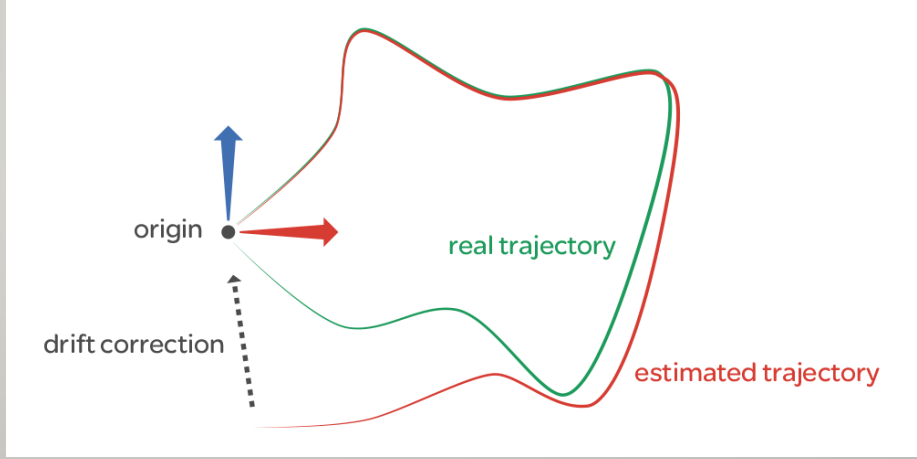
- HoloStudio, a full-scale 3D modelling application by Microsoft with 3D print compatibility
  - https://youtu.be/BRIJG0x_We8

Project Tango

- Initiate by Google special for AR on mobile device
  - https://get.google.com/tango/
- Motion Tracking
  - visual-inertial odometry
- Area Learning
  - Recognize and learn an environment
  - Improving the trajectory

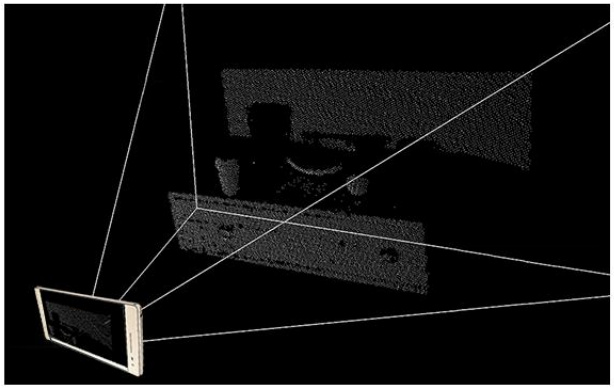
    Drift correction





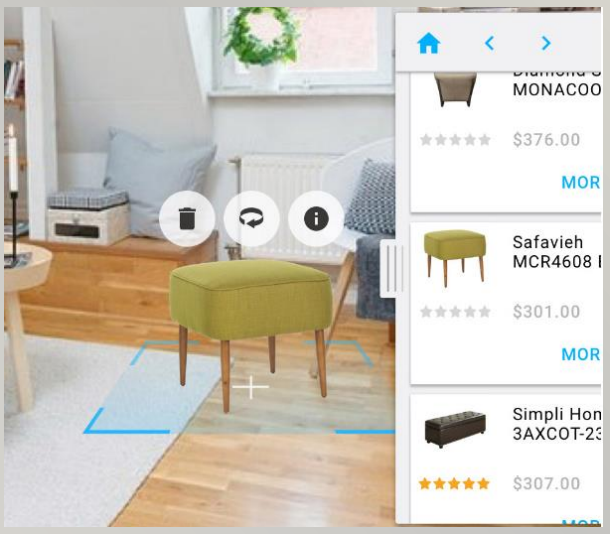
Project Tango

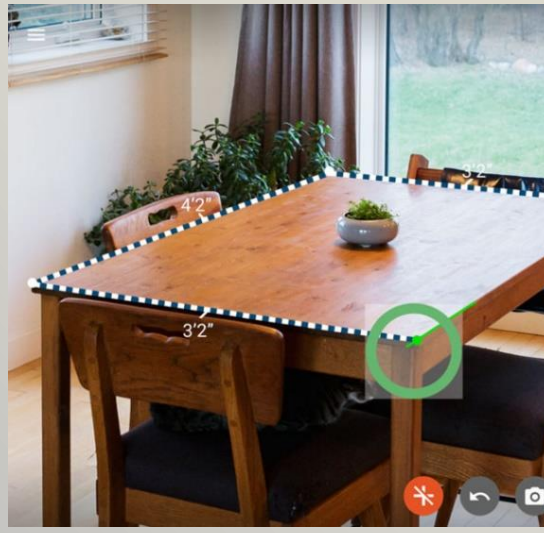
- Area Learning
  - Define Area Description Format (ADF) for pre-recording scene
- Depth Perception
  - IR (Structured Light, Time of Flight),
  - Stereo
  - So, it depends on device capability

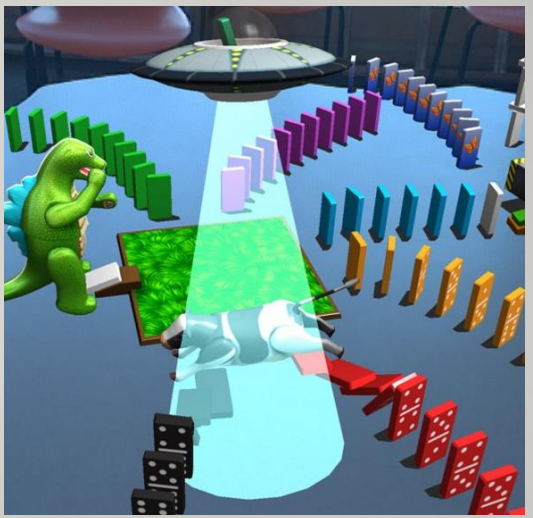


Applications

- Design and Measurement
  - Measure
  - WayfairView
  - Lowe's
- Education & Games
  - Solar Explorer
  - Domino World







- Check out https://get.google.com/tango/apps/

Summary

- From 2D to 3D graphics, the advancement of graphics supported in web browsers is fast in the last 2 decades
- With the emergency of WebGL, theoretically speaking any real-time rendering application can be run natively in browser
- Core techniques of marker based augmented reality are discussed
- Markerless ways for AR is emerging, technologies include Visual SLAM, SfM
- More MR related hardwares will appear, e.g. Hololens