3D Graphics Rendering, scanning, VR software

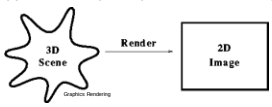
SGN-5406 Virtual Reality
Autumn 2009
ismo.rakkolainen@tut.fi

Rendering systems

- OHJ-2700 Computer Graphics
- http://www.realtimerendering.com/
- Conference papers: http://trowley.org/
- Generate signals from 3D models to output devices
  - Transform computer representations of the virtual world into real-time images sent to the display devices
  - Visual, aural, haptic rendering
  - 30 fps = VR engine needs to calculate frame in 33 ms
- Challenges: speed, stability, user interaction, hardware limitations
- Hardware and software systems

Visual rendering systems

- The process of converting the 3D geometrical models of a virtual world into a 2D scene presented to the user
  - The conversion of a **scene** into an **image**
- Software rendering system
  - Graphical rendering routines and formats
  - Parses a file containing prebuilt graphical shapes and/or instructions to generate the shapes that compose the image
- Hardware for rendering the graphics
  - Display adapter
  - Graphics processor (GPU)
  - Hardware support for many complicated 3D drawing operations & effects



CG History

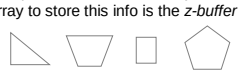
- Vector vs. raster graphics
- Sutherland Sketchpad demo
  - http://www.youtube.com/watch?v=USyoT_Ha_bA
- Douglas Engelbart : The Mother of All Demos
  - http://www.youtube.com/watch?v=JfIgzSoTMOs
- CG: Charles Csuri, Robert Abel
- Computer animation from 1975
  - http://www.youtube.com/watch?v=wduTjxYCunI
- 1981 Early Computer Graphics
  - http://www.youtube.com/watch?v=CK7b7oc7hWI

Illumination and Rendering

- To paint the picture, figure out the color and intensity of light hitting each point in the image. The whole problem is very difficult
- **Photorealistic image synthesis**: synthetic pictures that are indistinguishable from real photographs
- **Physically based rendering**: model the light the way light interacts with the environment
- **Reflectance**: the fraction of incoming light that is reflected
- **Transmittance**: the fraction of incoming light that is transmitted into the material. *Opaque materials* have zero transmittance
- **Diffuse**: The component of illumination which is reflected in all directions from a point on the surface of an object. Does not depend on the observers position relative to the surface point
- **Specular**: The component of illumination which is produced by reflection about the surface normal. It depends on the observers position. Appears as a "highlight" at shiny surfaces
- **Ambient**: The amount of illumination which is assumed to come from any direction and is thus independent of the presence of objects, the viewer position, or actual light sources in the scene

Polygonal rendering

- The simplest method
- Polygons are planar shapes defined by line segments
  - Normally 3 or 4 edges
  - Usually polygons are organized as polygon meshes for efficiency
- Most of real-time graphics is rendered using this method
  - Graphics hardware almost exclusively use this method
  - Fast algorithms and special effects for polygonal rendering integrated into hardware geometry engines
  - Z-buffer: iterate over objects, drawing each one as a polygonal surface, and for each pixel keep track of the closest object that we have seen so far. The array to store this info is the *z-buffer*



Polygonal rendering

- Vertex
  - A corner point of a polygon
  - Shared by neighbor polygons

Dino with 1558 polygons

Dino with 638 polygons

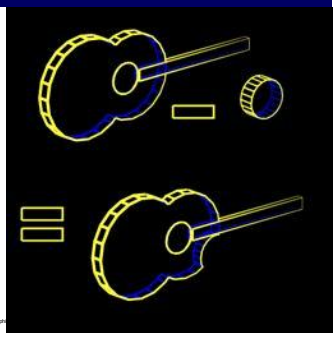
Geometrically based rendering systems

- Non-uniform rational B-splines (NURBS)
  - Parametrically defined shapes
  - Can be used to describe curved objects
- Octrees
- Fractal geometry methods
- Constructive solid geometry (CSG)
  - CSG objects are created by adding and subtracting simple shapes together
    - Spheres, cylinders, cubes, etc.
    - For example, a table can be created by adding five parallelepipeds, four as legs and one as tabletop

Geometrically based rendering systems

CSG example:

- Here cylinders and a box have been added together
- Another cylinder is then subtracted from the model to make a simple guitar model



Nongeometric rendering systems

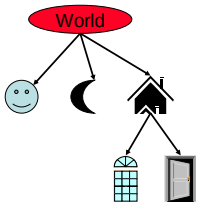
- Volume rendering
  - Well suited for semitransparent objects
  - 3D pixels, voxels
    - E.g., medical applications
  - Commonly accomplished using ray-tracing
    - Light sources defined
    - Behavior of the light rays based on the laws of physics calculated by the computer

Particle rendering

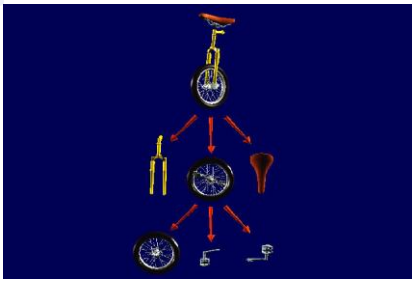
- Many small particles are rendered over time
- Produces visual features that reveal the process of larger phenomenon
- Often used to show complex flow in virtual scene
- Used in…
  - Fire
  - Explosions
  - Smoke
  - Water flow
  - Animal group behavior etc…

Scene Graph

- Description of the current state of the world
- Hierarchical organization of objects
- Can change over time
  - Grouping nodes
    - Hierarchical transformation tree
    - Cumulating:
      - Hand moves, fingers follow
  - Leafs
    - End a tree



Scene Graph



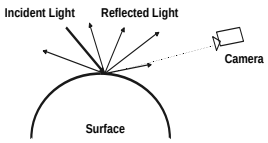
The rendering equation

$$L_o(x,\vec{w}) = L_e(x,\vec{w}) + \int_\Omega f_r(x,\vec{w}',\vec{w}) L_i(x,\vec{w}')(\vec{w}' \cdot \vec{n}) d\vec{w}'$$

- At a particular position and direction, the outgoing light (L_o) is the sum of the emitted light (L_e) and the reflected light. The reflected light being the sum of the incoming light (L_i) from all directions, multiplied by the surface reflection and incoming angle
- The whole overall movement of light in a scene
- http://en.wikipedia.org/wiki/Rendering_equation

Surface Reflection

- Light hits the surface from various directions. Some gets absorbed, some refracted, the rest is reflected. (let's assume everything is opaque from now on)
- If nothing is on the way, part of the reflected light will reach the camera, and form the image
- Different materials reflect light in different (and complicated ways) which is why they look different. We see them different
  - The details of reflection depend on the surface's microstructure

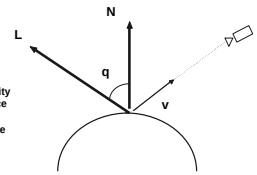


Lambertian (Diffuse) Reflection

- The simplest kind of reflector to model is an *ideal diffuse reflector*, also called a Lambertian reflector or chalk
- Incoming light is scattered equally in all directions, so brightness doesn't depend on view direction
- Reflected brightness depends on the direction and brightness of illumination (incoming light). This is given by the cosine of light/normal angle

$I = I_p\, k_d \cos q$

I_p: Light source intensity
k_d: Surface reflectance (bet. 0 and 1)
q : Light / Normal Angle

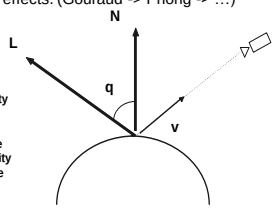


Basic Shading Models

- Historically, shading in computer graphics and VR started with the ultimately simple Lambertian model (ideal diffuser) and more and more terms were gradually added in to account for additional effects. (Gouraud -> Phong -> …)

$I = I_a\, k_a + I_p\, k_d\, N.L$

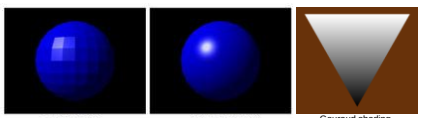
I_p: Light source intensity
k_d: Surface reflectance (bet. 0 and 1)
q : Light / Normal Angle
I_a: Ambient light intensity
k_a : Ambient reflectance (bet. 0 and 1)



The Lambertian model, with an ambient light term added in

Simple shading methods

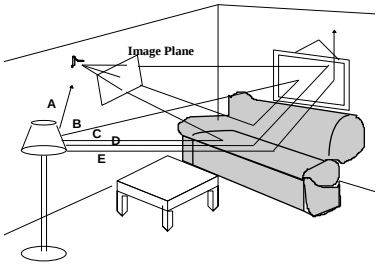
- Flat shading for Lambertian surfaces
- Phong shading for specular highlights
- Gouraud shading makes polygons look softer



Ray tracing

- Photo-realistic images of nearly any scene
  - Shadows, reflections, specular highlights, textures
- Global illumination based rendering method
- A point sampling algorithm
  - Sample a continuous image in world coordinates by shooting one or more rays from the eye through each pixel
    - leads to aliasing: anti-aliasing needed
  - The rays are tested against all objects in the scene to determine if they intersect any objects
  - If we hit a light source, we're done
  - If we hit a surface, we ask where the light impinging at *that* surface came from by shooting out more rays recursively
  - If the ray misses all objects, then that pixel is shaded the background color

Ray tracing



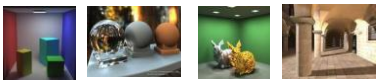
Some rays (like A and E) never reach the image plane at all. Others follow simple or complicated routes.

Ray tracing

- Ray types
  - **Eye rays** bring light directly to the image (pixel rays)
  - **Illumination rays** bring light directly from a light source to an object surface (shadow rays)
  - **Reflection rays** carry light reflected from a surface
  - **Transparency rays** carry light passing through an object
- A lot of computation needed, often not real-time
  - Each ray must be checked against all objects
  - Each ray may generate much more rays
  - Bad for VR, if too much calculation
  - Several approaches to speeding up computations:
    - 1. Use faster machines
    - 2. Use specialized hardware, especially parallel processors
    - 3. Speed up computation by using more efficient algorithms
    - 4. Reduce the number of ray - object computations
- E.g., PovRay software

Radiosity

- Attempts to simulate the complex inter-reflections of illuminated surfaces. Good especially for static and diffuse environments
  - Precomputed illumination!
  - Well suited for VR!
  - Unfortunately few environments are purely diffuse and static (especially in dynamic VR) -> progressive radiosity etc.
- Ray tracing and radiosity are complementary photorealistic techniques => hybrids, improvements
- Cornell radiosity box:
  - http://www.graphics.cornell.edu/online/box/compare.html



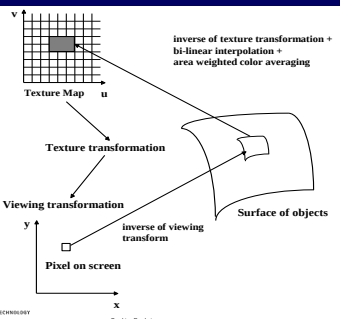
Other imaging methods

- Light field rendering, computational imaging, Lumigraphs, multi-camera arrays
  - http://www-graphics.stanford.edu/projects/lightfield/
- Image-based rendering
  - Warping image sets
  - http://cs.unc.edu/~ibr/
- Spherical panoramas & videos
  - QuickTime VR, etc.
- Special models for natural and synthetic objects
  - Waves, plants, humans, clouds,
- Non-photorealistic rendering

Texture Mapping

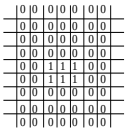
- Uniformly colored, flat shaded surfaces are boring and rare
- Real objects invariably have textures
- We can scan textures in from the real world (e.g., wood grain) or paint them, or generate them algorithmically, and store them in 2D image arrays
- Once we have the image, we map the (u, v) coordinates of a parametric surface onto the indices of the image array. During rendering, each time we sample the surface and compute a color, we fetch the appropriate value from the map
- This technique is called *parametric texture mapping*
- It has no great overhead, except for *anti-aliasing* texture-mapped images
- Increases realism significantly

Texture Mapping (cont.)

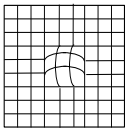


Bump Mapping

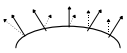
- Basic texture maps just paint things onto a smooth surface, but bump maps make a *rough* textured surface
  - We *could* add millions of little polygons with their vertices jiggled around
  - Instead, we can just *perturb the normal vector* at each surface point before the shading calculation. The surface geometry doesn't change, but we use shading to make it appear that it does



Bump Map



Bump Mapped Surface
(Appearance)



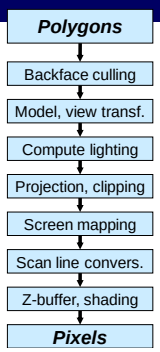
Perturbing Surface Normals

Environment Maps

- Paint six images on the faces of a cube, to make a "background" image that surrounds your objects. This could be clouds, sky, mountains, or a room
- When you do the shading calculation, use the normal and view vectors to figure out the light direction that would produce a specular ray to the viewer
- Intersect that light direction with the environment map
- Get the color from the environment map, and add in a light source with that color
- Result: you see an (approximate) image of the environment reflected from shiny surfaces
- A very effective and cheap hack that's widely used as an alternative to ray-tracing

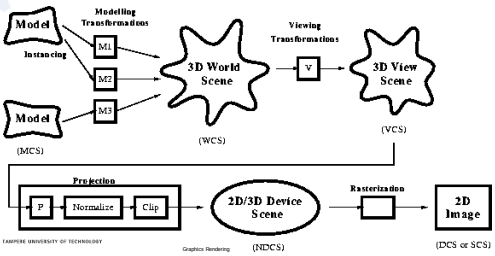
Graphics rendering pipeline

- Classically, "model" to "scene" to "image" conversion broken into finer steps, called the **graphics pipeline**
- Commonly implemented in graphics hardware to get interactive speeds
- Frame rate VERY important for VR, ~30fps
- Stereoscopic rendering requires suitable graphics cards with dual output
  - sequence of alternating left and right images displayed on a monitor/HMD
  - time-sequential, field-sequential, frame-sequential, or image-sequential
  - http://3d.curtin.edu.au/3D-PC/index.html



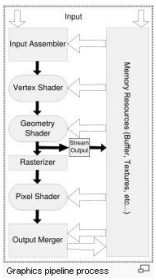
Graphics rendering pipeline

- At a high level, the graphics pipeline usually looks like this



Graphics rendering pipeline

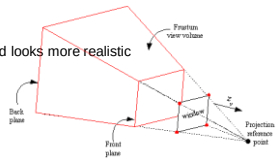
- Application stage
  - Done entirely in software by the CPU
  - Reads the world geometry database
  - Reads the user's input
    - Mice, keyboard, trackers, gloves etc.
  - Changes the view to the simulation or orientation of virtual objects in response to the user input
  - Simulations that drives the application
    - Physics, AI, collision detection, animation



Graphics pipeline process

Graphics rendering pipeline

- Geometry stage
  - Implemented in hardware (or software)
  - Model transformations
    - Translation, rotation, scaling etc.
  - Lighting computations
    - Calculate the surface color based on
      - The type and number of light sources
      - The lighting model
      - The surface material properties
      - Atmospheric effects (fog etc.)
    - The scene will be shaded and looks more realistic
  - Scene projection
  - Clipping
  - Mapping

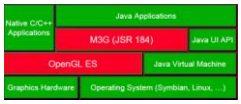


Display adapter

- Display, rasterizer stage
  - Vertex info to display pixels
  - A small specialized computer inside the computer, consisting
    - The processor for graphics, GPU (Graphics Processing Unit)
    - Memory, memory bus, motherboard connection bus (PCI Express)
    - Monitor connectors (Analog: VGA, S-Video. Digital: DVI, HDMI, DisplayPort)
- The graphics are processed in graphics processing unit (GPU)
  - Typically situated on display adapter card
  - Integrated graphics solutions
  - High-end cards used to cost $100,000+ only 10 years ago!
  - Largest manufacturers are NVIDIA and ATI
    - Powerful GPUs at reasonable prices (also Intel, Matrox etc)
    - 3D-capability emerging also to smartphones
  - GPU can be used also for many other calculations
- Upcoming AMD Eyefinity: 6 displays per card
  - Multiple cards, upto 268 megapixels

3D Graphics Libraries

- Programming 3D graphics in low level is done using some graphics library
  - OpenGL, ES
  - Direct3D
    - Part of DirectX
    - PCs, also for Xbox 360
  - M3G, Mobile 3D Graphics API
    - An API for Java programs that produce 3D computer graphics
    - Extends the capabilities of the Java Platform, Micro Edition for mobile phones etc.
    - High level 3D API, but may be slow (often no floating point unit)



- Modern GPUs support DirectX and OpenGL standards
- http://www.graphicsformasses.com/
- http://people.csail.mit.edu/kapu/EG_09_MGW/

OpenGL

- Hides the differing capabilities of hardware and the complexities of interfacing with different 3D accelerators through a uniform API. Converts primitives into pixels
- OpenGL 3.1
  - Released in 2009, backward compatible
- OpenGL ES
  - Subset of OpenGL, especially for mobile devices
- OpenGL Shading Language (GLSL)
  - More direct control of the graphics pipeline without assembly or hardware-specific languages
- Managed by Khronos consortium
  - http://www.khronos.org/
  - opengl-redbook.com
  - http://www.glprogramming.com/red/
  - OpenGL tutorial: http://nehe.gamedev.net/

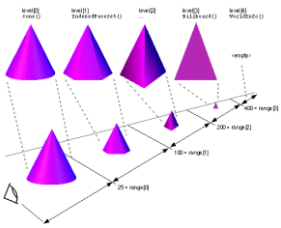
Display adapter bus

- PCI
  - Some years ago the display cards were connected to a standard PCI slot
  - Very slow interface, would not be suitable for modern graphics
- AGP
  - Now replaced by PCI Express
- PCI Express, 2.0, 3.0
  - Faster than AGP, has replaced AGP

- Direct Transport Compositor™ by Scalable Graphics is a middleware that enables any number of graphics cards and processors to concentrate their effort on a single display
  - No modification to the application
  - Higher frame rates and anti-aliasing
  - Stereovision support
  - http://www.scalablegraphics.com/

Level of Detail

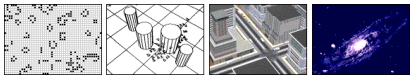
- Level of Detail (LOD): a usual trick or approximation to increase rendering speed
- Use as low level a 3D model as possible, depending on distance
- Billboards are textures which replace polygons alltogether. They turn towards the viewpoint



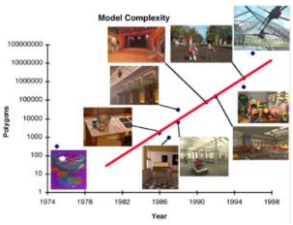
Optimization

- Performance affected by e.g.,
  - Number of lights, vertices, pixel sizes
- Bottlenecks
  - 1. Locate the bottlenecks
  - 2. Remove them
  - 3. If needed, Goto 1
    - If Application stage takes 30ms, Geometry stage takes 20ms, Rasterizer stage takes 40ms
    - The maximum framerate is then 1/40ms=25Hz
    - Optimizing the application stage won't make the app go any faster
    - Could get more time for doing more complex lighting equations (20ms idle time in Geometry stage!)
- Increase performance by
  - ad hoc formulations (fake it), low-res, HW, precomputation
  - Reducing the amount of data, disabling lights, fog, blending
  - Resizing window, reduce the number/size of textures, using LOD, billboards
  - Setup a number of tests, where each test only affects one stage at a time

3D Rendering



- Much more realism & speed, lower price in recent years
- Much bigger polygon models and data sets

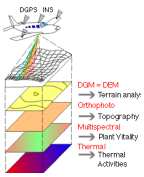


3D Modeling

- Many ways to acquire 3D data & to create 3D models
- Usually laborous, a bottleneck
  - Geometry, textures, others?
- Semiautomatic ways to create models?
  - Send images, get 3D model: http://3dsee.net/
- Utah teapot is a classic 3D model since the 1970's
  - http://www.sjbaker.org/teapot/

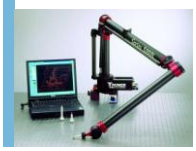
3D Modeling

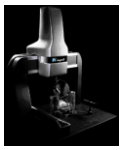
- CAD, 3D modeling software
  - 3D Studio MAX, Alias|Maya, Houdini, SolidWorks, LightWave3D
  - AutoCAD, CATIA, Presagis, Multigen, Cinema 4D, XSI etc.
  - http://www.blender3d.org/ www.students.autodesk.com
  - http://sketchup.google.com/
  - 3D editors in VR toolkits
- Existing 3D models
  - Importing CAD files (2D, 3D)
  - 3D object databases
    - 3dtotal.com. Smithmicro Poser 8
  - GIS, aerial stereo photographs
  - Google Earth, 3D warehouse
- 3D digitizers & scanners
- Other methods
  - Flat-land Inc.: 3DML (easy)
  - ATT's WordsEye: speech description



3D Digitizers

- Measurement Arms (Faro, Romer etc.)
- 3D pantographs (Microscribe, Hyperspace etc.)
- Tactile robots
- CMM (Coordinate Measuring Machine)
  - Separate points, even microns
  - Based on touch, laborous, slow, expensive





Optical 3D Scanners

- Fast way to create models
- Laser range scanners and/or computer vision
  - Usually structured light / laser, which makes things easier
  - Sometimes no emitted light, but images only
    - Much harder computationally
  - Problems are glass, dark, and glossy surfaces
  - Polhemus FastScan
  - Cyrax 2500
  - DAVID-Laserscanner: freeware for 3D laser scanning
    - http://www.david-laserscanner.com/
- 3D desktop scanning, $500
  - http://www.real-view3d.com/
- Illumination capturing, Aguru dome
  - http://www.aguruimages.com/

3D Scanners

- Many discontinued products
- Microscribe, 3500 - 7000 €
- Cyrax 2500, 150,000 €
- Polhemus Fastscan
- Freepoint3D
- http://www.inspeck.com/
- http://www.eyetronics.com/
- Cyberware
- 3Q, http://www.3q.com/
- http://www-2.cs.cmu.edu/~cil/v-hardware.html

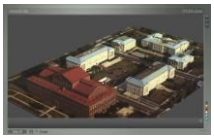
3D Digital Cameras

- E.g., 3D VuCam
  - http://www.stereovisioninc.com/

- Visionsense 3D medical camera
  - Helps on minimally invasive surgery

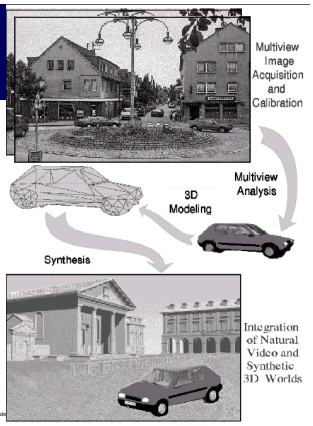
(Semi)automatic Modeling

- Canoma, Geometra, D Sculptor



- Minolta: 3D digital camera, 4000 €, discontinued
- 3D City Engine, semiautomatic city models
  - www.procedural.com

Images to Models



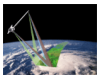
3D Scanning Procedure

- Data acquisition
  - multiple strategically located cameras, laser range scanners or such
- Calibration
- Surface registration
- Mesh reconstruction
- Data modeling and smoothing
- Texture reconstruction

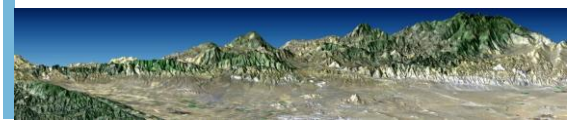
Virtualized Reality™

- Carnegie-Mellon University 2001
  - http://www.cs.cmu.edu/afs/cs/project/VirtualizedR/www/VirtualizedR.html
- Capture, analyze, model real events
- 3D + time = 4D digitization
- Bottlenecks:
  - Content creation, 3D-modeling, texturing

Space Shuttle Radar Topography Mission

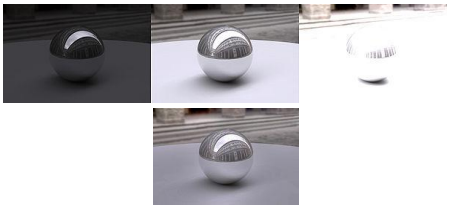


- Launched in 2000
- Radar interferometry
- Large scale Earth 3D modeling
- 80% of Earth's landmass (60°N-56°S)
- 12.3 Terabytes total data
- Digital topographic map
  - 30 m x 30 m spatial sampling
  - <=16 m absolute vertical height accuracy



High dynamic range imaging

- High dynamic range imaging allows a greater dynamic range of luminance between light and dark areas of a scene



Spherical Imaging etc.

- Panoramic, spherical images and videos provide a way for look-around and telepresence applications. Limited moving may be possible
- Immersive Media Corp., a leader in 360º spherical video technology, http://demos.immersivemedia.com
- Panoramic cameras: fisheye, http://immervision.com/en/home/index.php
- D7, stitch, no camera movement, http://scallopimaging.com/
- Multiple positional IR sensors, moving camera, http://luciddimensions.com/
- Spherical video, Point Grey Labybug 3, http://www.ptgrey.com/
- Softkinetic 3D cameras, http://www.softkinetic.net
- Lots of panoramic pictures: http://www.vrmag.org/, http://www.gregdowning.com/, www.world-heritage-tour.org
- Google's Street View with panoramic streetviews
- Microsoft Photosynth takes regular photographs and reconstructs the scene or object in a 3D environment. Share them with friends. http://photosynth.net/
- Microsoft research HD View Gigapixel Panoramas, http://research.microsoft.com/en-us/um/redmond/groups/ivm/HDView/

Computational Photography

- Video or still camera arrays for improved imaging, acquisition, special effects
  - Artificial focus, super-resolution, etc.
  - Adobe 3D lens
  - Liquid lenses (rensselaer poly), fresnel, etc.
  - Origami optics, adaptive optics (astronomy)

Raskar, Tumblin: Computational Photography
- computationalphotography.org
- Course: http://web.media.mit.edu/~raskar/photo/2008

Plenoptic Cameras

- Integral imaging
  - Lens arrays capture & display the 3D image
- E.g., Stanford multi-aperture camera
  - Slightly overlapping 16x16 subarrays
  - Each subarray has its own microlens
  - 3D depth computationally
  - Can reduce noise, super-closeups
  - Lens: lithography, semiconductor processes
  - Disadvantages
    - Lower resolution
    - Power consumption, 10x calculation
    - Requires textures

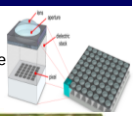


Image in focal Plane

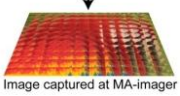


Image captured at MA-imager

Kinematics Modeling

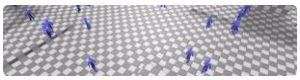
- Location, rotation and motion in 3D world
  - Kinematic control: moving objects without forces
  - Dynamic control: calculate forces to produce the motions
- Skeleton, muscle, skin layers
- Parent-child hierarchical relations (body, hand, finger)
- Direct kinematics
  - Specifying rotations all the way from root node to leaf node to get the leaf nodes position and rotation
- Inverse kinematics (IK)
  - The opposite of direct kinematics, heavily used in robotics
  - Position/rotation of the leaf node is known, calculate all rotations up to the root node
  - Used in interaction, motion capture, modeling tools
  - Closed or numerical solutions
- Together IK and a hierarchical skeleton enable specific motions in real-time

Physical Modeling

- Gravitation, friction
- Collision detection
- Collision detection software:
  - I-COLLIDE, SWIFT++, OPCODE, AABB trees, SOLID
- Collision, surface deformation, force computations
  - Elasticity, collisions etc.
  - Force feedback
  - Vortex™, Novodex™, Havok ™, VPS-PBM™ , Renderware Physics™ , Meeqon Physics SDK

Behavior Modeling

- Human or crowd behavior modeling
  - Virtual humans (avatar = 3D model of an online user)
- Autonomic objects, intelligent Ves, realistic scenarios
- Applied e.g. in games
- Methods: scripting, fuzzy logic, neural networks, ...
- Simulation of emotions
- Artificial intelligence
  - Presagis AI.implant is an AI authoring and runtime software
  - Simulated characters make sophisticated context specific decisions
  - Authoring the AI world, creating rules used for decision making
  - ViCrowd, PeopleShop, MRE, games



VR Toolkits

- Commercial VR software and hardware: e.g., http://www.est-kl.de/
- List of VR software: http://www.diverse.vt.edu/VRSoftwareList.html
- Low-level programming (e.g. OpenGL) would be possible, but hard
- WorldToolKit® is a multi-platform software development system for building high-performance, real-time, integrated 3D applications for scientific and commercial use
  - A set of API, function libraries and tools to create, prototype, develop, configure, manage and commercialize applications
  - Support for networked distributed simulations
  - Options for immersive displays, and I/O devices (HMDs, trackers, etc)
- 3DVIA Virtools, many plugins
- The Lightning VR system can be used either as a development system or as a runtime module under Unix/Linux
  - Sets of functions for 3D models, I/O and communications. API allows the user to make adjustments using either the incorporated language or use C/C++ modules even during run-time
- OpenGL Performer™ is a programming interface helping developers to create real-time simulation 3D graphics applications
  - Capabilities that help in complex applications such as VR, scientific visualization, interactivity and CAD. Incorporates capabilities of using multi-processor and multi-graphics pipelines. Built on top of the OpenGL® standard graphics library and combines Ansi C and C++

VR Toolkits

- WorldViz Vizard
- OpenSG
- Alice 3 beta software, http://www.alice.org/
  - (VR professor Randy Pausch's "Last Lecture": youtube.com/watch?v=ji5_MqicxSo)
- CAVElib is a C programming interface that can be combined with Performer, Open Inventor or other OpenGL renderer to manage VR systems and help building VR applications:
  - Display configurations, multi-pipe parallel rendering, stereo and tracking
- Input device interface libraries
  - A number of libraries are available to simplify application developer task to manage the various input devices in VR systems (commercial e.g., trackd, open source e.g., VRPN, OpenTracker)
- VR Juggler is an open-source VR application devl. framework
  - APIs that help the programmer on interface aspects, display surfaces, tracking, navigation, graphics and rendering techniques
  - The VR Juggler Suite includes a virtual application, a device management system (local or remote access to I/O devices), a standalone generic math library, a portable runtime, a simple sound abstraction, a distributed model view-controller implementation and an XML-based configuration system

VR Toolkits

- OpenSceneGraph
- DIVERSE is a cross-platform, open source API for developing VR on Linux or Windows. Enables to quickly build applications that will run on the desktop or immersive systems. DIVERSE can interact with other APIs and toolkits
- Collada is an initiative to create a portable 3D file format
- Maverik - VR micro kernel, http://directory.fsf.org/project/maverik/
- VRED, vred.org
- PeopleShop, PeoplePak
- Real time physical engines
  - Collisions, rigid body dynamics
  - Vortex™, Novodex™, Havok ™, VPS-PBM™
  - Open source: ODE (Open Dynamic Engine)
- Haptics libraries
  - Device-dependent, e.g. GHOST, ReachIn API,
  - Sensable FreeForm for modeling
  - Immersion VirtualHand® SDK
    - Development environments for adding hand-motion capture, hand-interaction and force feedback to simulation applications

VR Toolkits

- Open Inventor
- VRML / X3D libraries
- Java3D
- Alias MOCAP™ is a real-time motion capture system designed for the capture, editing, and blending of motion data
- Alias Online for broadcast productions
  - Integrates 3D animation and live video
  - 3D characters that interact with live actors in real time
- Alias MOTIONBUILDER™ to create 3D character animation
- The MotionMonitor is a real-time 3D motion capture system
  - Advanced data capture and analysis package
  - Designed for use in medical research; physical therapy clinics; sports medicine labs; motor control, balance, neurological, and gait studies; golf, tennis and baseball instruction
- A very large model visualization with Realityserver
  - www.mentalimages.com/realityserver
- Commercial & free VR packages
  - http://conferences.computer.org/vr/2009/IE3_tutorialMaterials.html