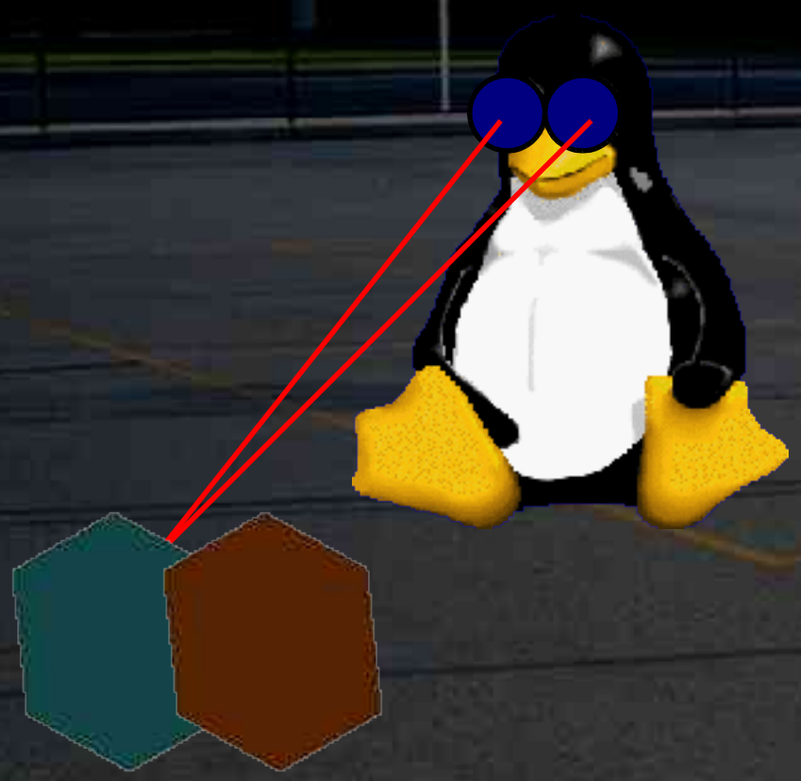
Hacking Your Own Virtual and Augmented Reality Apps for Fun and Profit!



Tutorial Presentation

Linux Conf Au - Adelaide, SA

January 2004

Wayne Piekarski

Wearable Computer Lab

University of South Australia

School of Computer and Information Science

Adelaide, South Australia

wayne@cs.unisa.edu.au

S34°48.640' E138°37.201'

What is this tutorial about?

- Brief introduction to 3D, virtual and augmented reality
- OpenGL and live video display under X11
- Video capture using Video4Linux and Firewire
- 3D vision tracking using ARToolKit
- Custom hardware input devices
- Tour of Tinmith-Endeavour backpack

- Show you the kinds of cool applications you can build at home without having to spend a lot of money!

What is this tutorial not about?

- Will not repeat what you can easily learn elsewhere
  - Linux distro installs, basic OpenGL, simple C programming
  - I will assume you know something about these
- Mainly focusing on Linux specific problems and solutions
  - Lots of things which are not documented very well

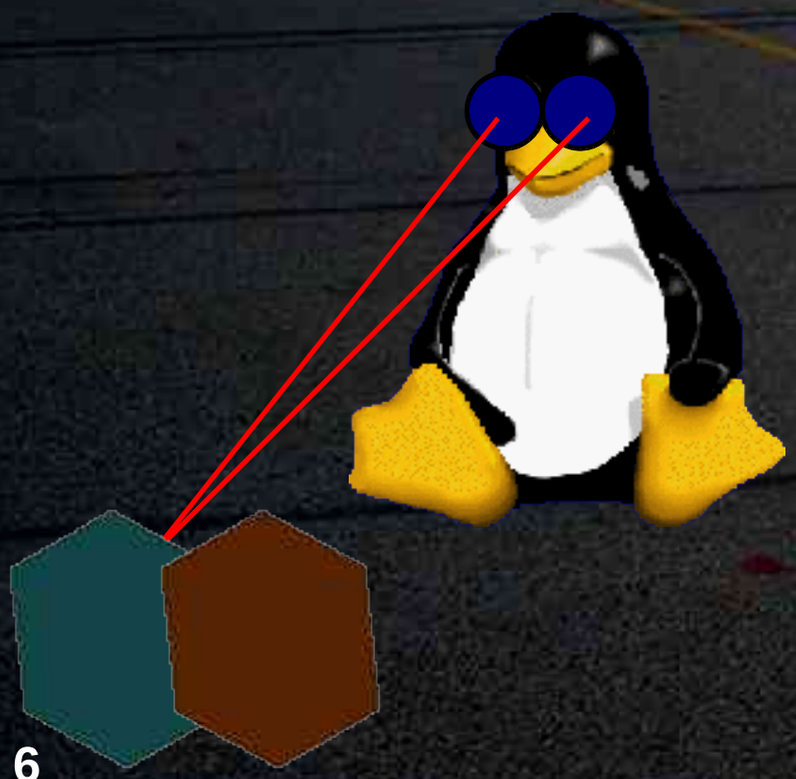
Tutorial and supplied materials

- 3 hour presentation with a break half way
  - 1.5h talking, 0.5h break, 1.5h talking
  - Question times at the end of each section
- 40 pages of notes with extra material and code snippets
  - This presentation will be more higher level than the notes
  - Will talk about things at a different angle than the notes
- CD contains example demos and scripts
- Also includes open source programs and libraries
- After the tutorial we will take the backpack outside
  - Inspection of internals, as well as demos

Lazy like a fox

- Teach you interesting things you can use at home to hack around with 3D right now!
- Lots of projects, not enough time to work on them
- Focus is on areas that are poorly documented or difficult to play around with due to complexity
- Tricks to build things on the cheap

Intro to immersive 3D graphics



Immersive 3D graphics

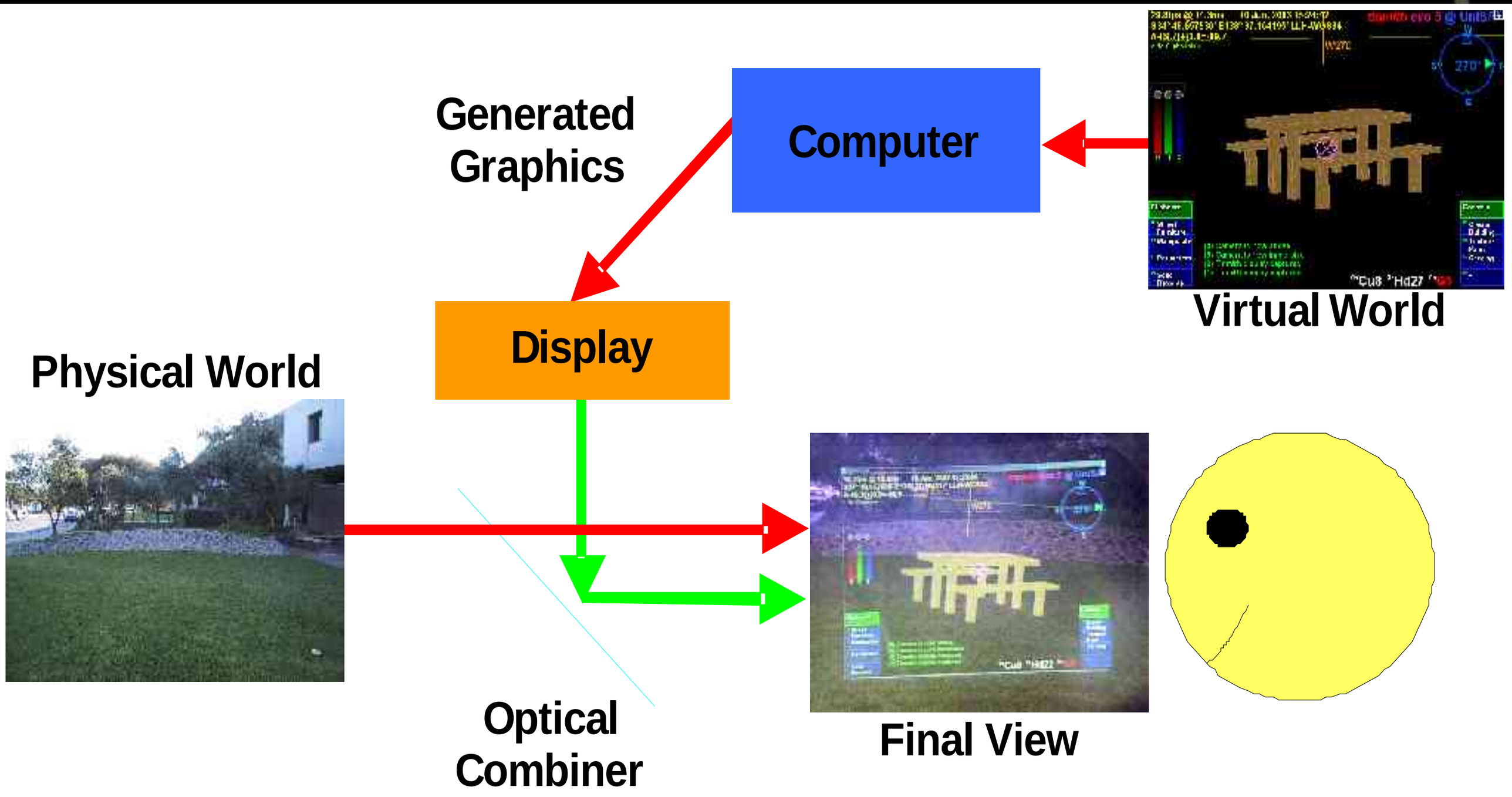
- My research work focuses on immersive 3D applications
- Typically use a head mounted display, not a monitor

Virtual and augmented reality

- Virtual reality is purely computer generated graphics
- Augmented reality combines the physical world with artificial computer graphics

Optical augmented reality

- Hard to find these displays for a reasonable price now



Video augmented reality

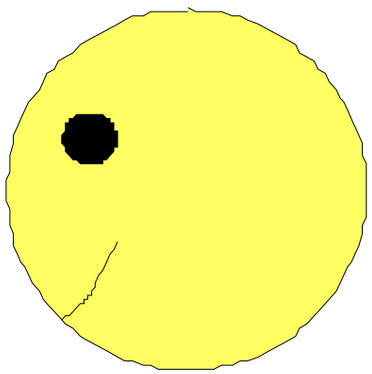
- Displays from VR can be used with no modifications
- We use these displays almost exclusively now
  - Cheaper and easier to buy, better quality output

Virtual World

Computer

Generated Graphics

Final View



Physical World

Camera

Display

Outdoor augmented reality

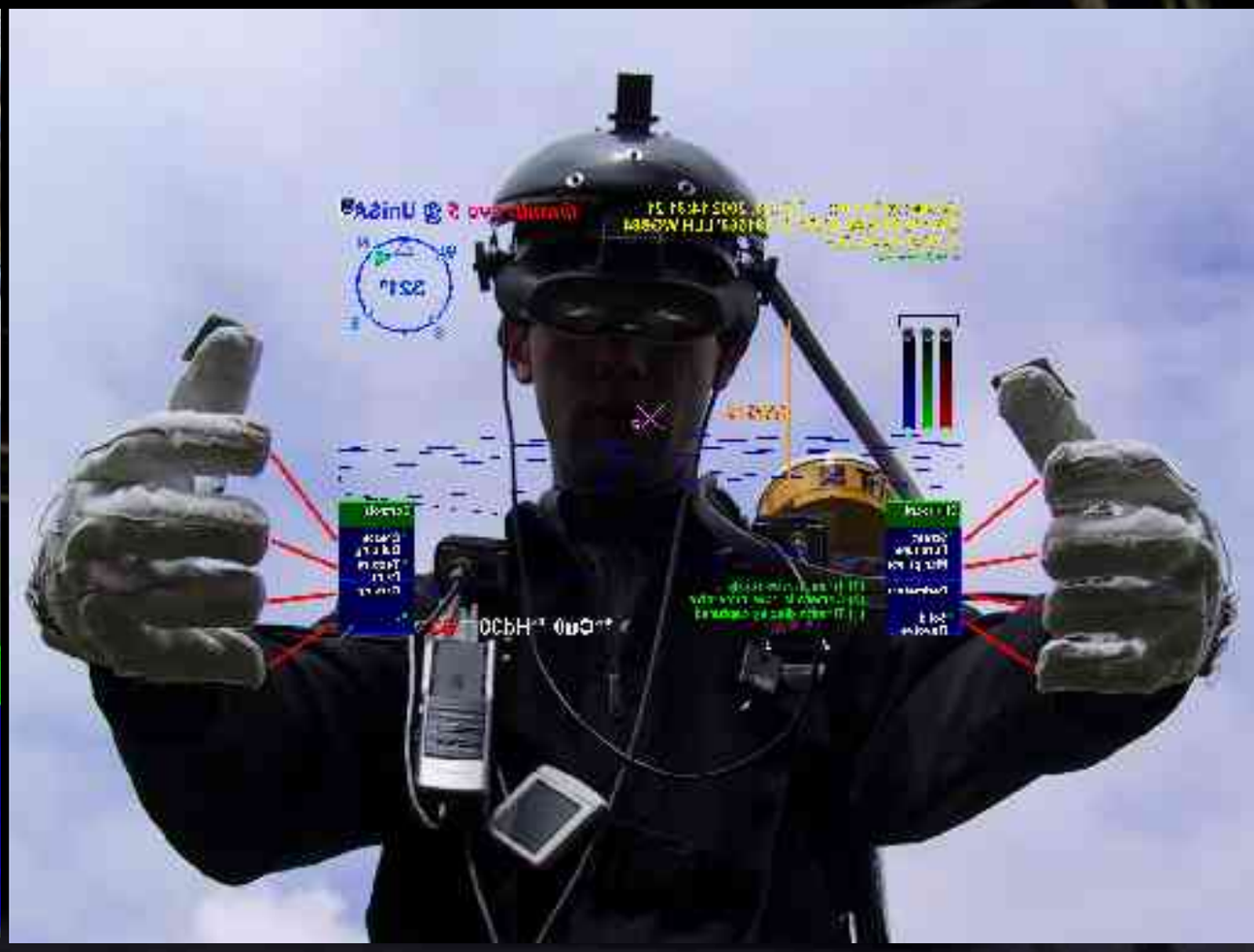
- My research work focuses on performing AR outdoors
- Especially mobile 3D user interfaces and modelling

3D compared to 2D

- 3D is much more challenging than 2D
  - More degrees of freedom and more input devices
  - More realistic and intuitive application possibilities
  - Potential to use the body directly
- Don’t use 2D input devices to solve a 3D task!

Tinmith mobile 3D modelling

- Gloves are used to control the environment
- User interface designed specifically for mobile AR
- Supports outdoor modelling and editing applications



User interface demonstration

- Play UI demo video here

ARQuake demonstration

- Play video here

Tracking

- To generate a view, the computer needs to know the position and orientation of the user's head
- May also require tracking of body parts and tools
- Restricts the types of user interfaces we can use!

- Accel/gyro/magnetic InertiaCube2
  - A$3500
- Trimble Ag132 GPS
  - A$8000
- ARToolKit vision tracking
  - Cost of video camera = $Cheap

- Magnetic, optical, ultrasonic, mechanica

Processing

- We need to carry sufficient computing power with us
  - 3D graphics requires fast video chipset
  - Video processing and capture can be CPU intensive
- Most small computers have stripped down graphics hardware because the demand for this is low
- Laptops used to be pretty poor as well
  - Games market spawned powerful laptops with 3D GPUs
- Getting laptops with good 3D and small sizes and good power consumption is still a problem though

Indoor AR/VR

- Not everyone has a backpack like me

- Indoor setups for VR and AR
  - Still require expensive tracking hardware and HMD
  - Slightly easier because hardware can be bulky
  - Limited operating area

Free beer AR

- AR and VR are dominated by the cost of the hardware
- But free beer and free speech are also possible in AR!
- ARToolKit
  - Hirokazu Kato and Mark Billinghurst
  - University of Washington
- More on this later
- Show live demo

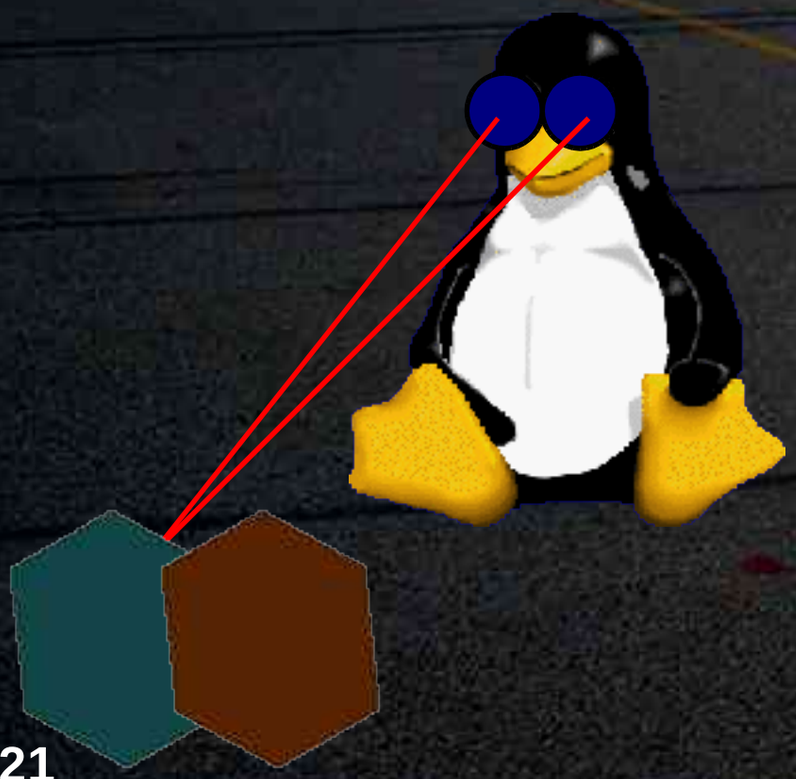
Tutorial discussion

- Today we will talk about how to use
  - Video cameras
  - ARToolkit
  - 3D renderers
  - Custom hardware

- Goal is to allow creation of 3D apps to run on a desktop
  - What else is possible apart from the standard 2D UI?

- Lets get started!

Distribution installation



Distribution installation

- I use Knoppix as my development platform
  - Easy to setup and comes preconfigured for most things
  - I found Debian too hard to do very simple things
  - Has all the advantages of Debian though
  - Based on Debian unstable so it is up to date

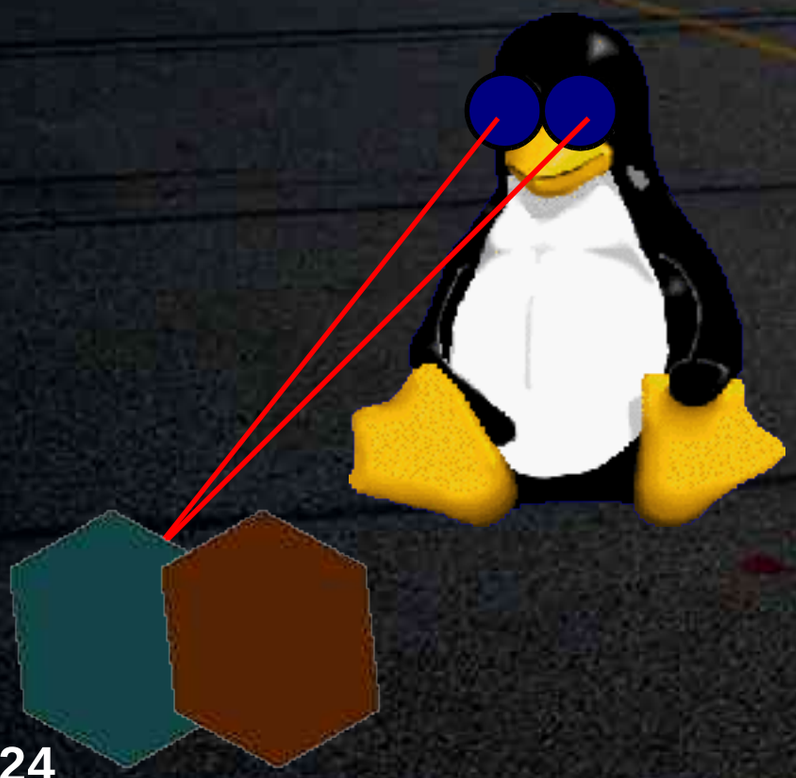
- Problems with Knoppix
  - Unstable and testing have problems
  - Debian stable is not really up to date enough either

- Still looking for an ideal distribution
  - I can’t afford to spend weeks configuring a machine
  - Simple things should be simple, complex to be possible

Software requirements

- Does not matter what distribution you use
  - To be on the safe side, install as much as you can so you can avoid chasing up missing packages
- Make sure you include (see notes)
  - Devel, XFree86, DRI, OpenGL, Kernel, Firewire
- Knoppix has almost everything
  - Use apt-get to install the following
  - coriander, libraw1394-5, libraw1394-dev, libdc1394-dev, libglut3, libglut3-dev
  - The conference LiveCD will hopefully have these too

3D graphics infrastructure



Graphics libraries

- Xlib
  - Existing X11 drawing library used for all X applications
- GLX
  - Extension to X11 to allow OpenGL over X connection
- DRI
  - Direct rendering interface for XFree86
- GL
  - Core drawing functionality, possibly accelerated
- GLU
  - Utility library, simplifies some GL calls for the programmer
- GLUT
  - Utility toolkit provides a portable programming interface
  - Allows apps to run on almost any system

OpenGL advantages

- Previously, required both X server and client processes
- For intensive drawing you waste a lot of CPU time
  - X protocol and memory usage
  - Kernel read() and write() calls
  - Task switching
  - X is not very good at massive primitive numbers
- This is why SGI implemented direct rendering
- Now we have GLX and DRI in XFree86 based on this

- Server sets up video card for security
- Client can then run without any intervention
- System calls are evil, we can avoid them now!

3D hardware is cheap

- Nowadays 3D hardware is available in almost all PCs
- Old cards are still excellent, get them for free from people throwing them away!
- Older cards have more mature drivers as well

- 3D drivers are very complicated monsters!

- Most cards do not use hardware for all features, only the most commonly used ones
- SGI used to implement full hardware support though

Nvidia and ATI

- Nvidia provide well supported closed source binaries
  - TNT2 is reasonable, GeForce2 is minimum recommended
  - Used this extensively on many boxes with good results
  - Same code base as Windows so has good performance

- DRI provide open source driver for ATI cards up to 9200
  - Works ok but still has bugs in process of being fixed
  - Use it on my laptop, its ok but the Nvidia is more polished
  - Future for DRI is not certain past 9200?
  - ATI provide binary drivers but not as good as Nvidia
  - Minimum recommended is Radeon 7500

Nvidia and ATI

- Avoid other hardware if you can
  - Integrated chipsets are much slower
  - Nvidia and ATI are the two big players
  - Drivers tend to be not as good for less mainstream gear
  - A problem if you use a laptop however!

- Nvidia versus ATI always starts religious flame wars
  - Nvidia do not release specifications for their cards, but they do provide good quality drivers that work
  - ATI DRI are getting better, but still have some issues

- ATI are more power and size efficient for laptops

Learning OpenGL

- OpenGL Red Book is what everyone learns from
- Many free tutorials available on the web

- OpenGL is beautifully designed and super easy to learn
  - It is not convoluted and tricky like Xlib
  - Simple things are simple
  - Optimisation and tricky things are still possible

- Supports primitive rendering, shading, textures, depth buffering, 2D and 3D projections, linear transformations, and display lists
  - Everything you need to write both 2D and 3D applications
  - X supports 2D primitives and windowing, that's about it

Using OpenGL

- I write most of my apps using pure OpenGL now
- I use X only as a way of opening a window for OpenGL

- No way under X to draw lots of primitives without incurring the overheads described before
  - If you use X then you get hardware acceleration but you are limited by client/server overheads
  - If you draw locally you can use SHM and DGA to copy direct to hardware but you have no primitive acceleration

- I've always wanted an Xlib which was compiled directly into the client or allowed direct access to the hardware
  - We now have it in the form of DRI

Catches of OpenGL

- Pipelined in video card to maximise performance
  - May introduce some latency, not sure how much
  - Doesn't seem to cause a problem for live video though
- OpenGL is just a renderer, whereas X has toolkits on top of it like Qt and GTK
  - I guess that some of these have been ported though

Useful example

- X has trouble displaying live video
  - MIT SHM, Xvideo, DGA, etc
  - Each driver in XFree86 supports some but not others
  - You have to write your client to support all of them!
  - X doesn't expose the entire acceleration of the video card

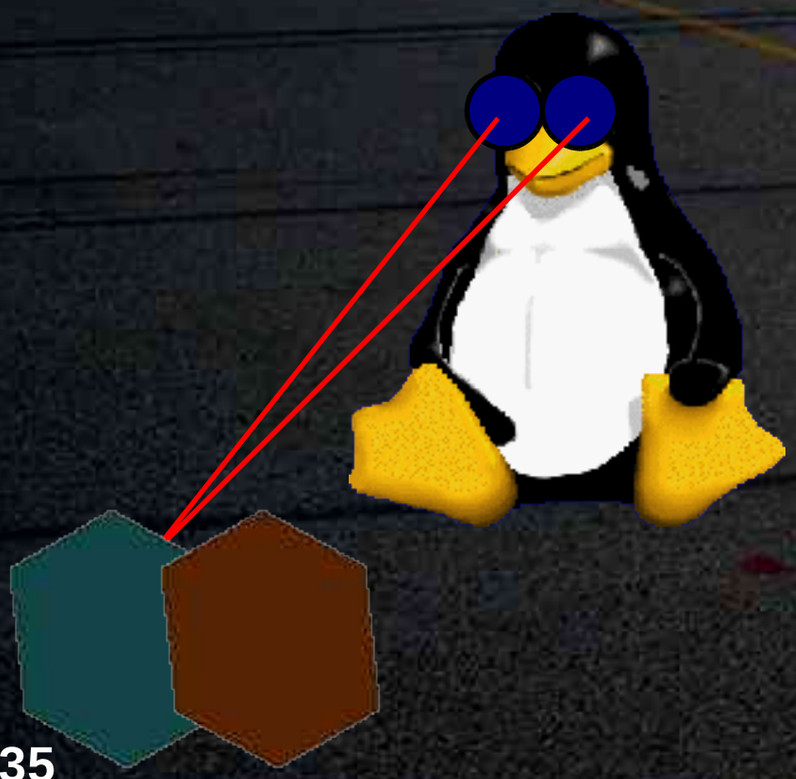
- OpenGL can display texture maps natively
  - Not just render but perform warping, scaling, etc
  - Can render to flat 2D or any 3D polygon!
  - OpenGL is written to assume acceleration, so it will take advantage of as much as is available
  - Supports auto format conversions (RGB, YUV, B&W)

  - Show application demo here

Video display source code

- Work through example code here

Live video capture



Capturing video

- Previous example showed how to display video
- Now we show how to grab this video

- Capturing video is supported under Linux
- Interfaces are non-trivial and tricky to use however
- Really needs a nice user land API to simplify it
- Developers have to supply a lot of their own code

- Not much documentation
- Not commonly used, so not maintained as much

Video 4 Linux

- V4L was the first common kernel API for capture drivers
  - PCI capture boards, USB cameras
  - Xawtv display program

- Simple interface calls
  - open() – initialise device
  - read() – read data from camera, can also use mmap()
  - ioctl() – configure capture settings

- You can specify resolution and pixel format (RGB, YUV) but the device must support it
- If not then you must supply your own conversion

Video 4 Linux

- If you have trouble using V4L, reload the modules
- I found the CPIA camera drive was not very reliable

- New V4L2 API is available in kernel 2.5 and 2.6
  - Not all drivers use this new API
  - Older V4L apps are supported via compatibility layer
  - Designed to fix some flaws in old V4L
  - Still needs a nice user land interface library

- Go through example V4L code

Firewire video capture

- Linux has had Firewire support for about two years
  - Hard drives
  - Digital video cameras (DV)
  - Digital cameras (DC)

- DV devices are supported using libDV
  - Takes raw DV compressed video from video camera

- DC devices are supported using libDC
  - Supports YUV and RGB raw video up to 640x480
  - Nice for PCs because no decompression required

- We will talk about DC devices today

libDC video capture

- The libDC library is complicated and has almost no documentation except for some include files
- The easiest way to capture from DC devices is to use the ARToolKit code because they have written an interface
  - ARToolKit is modular so you can use just the capture code
  - I never wrote my own interface because it is quite tricky
- The DC specification defines interface for all cameras
  - Very nice because ALL 1394 cameras work with Linux!
  - Contrast to USB cameras where there is no standard and very poor driver support
  - I have bet the farm on DC cameras, they are a bit more expensive but they are nice to use

Using DC devices

- The RGB24 mode is nice because you can take the raw image data and work with it straight away
  - Wastes bandwidth but CPU is not used much
  - USB cameras tend to operate in YUV mode

- Coriander is a nice program for playing with cameras
  - All devices have controllable settings which is nice

- I found that DC devices are more robust and reliable than the CPIA USB camera I used
  - I have used the Pyro Webcam and Point Gray Firefly

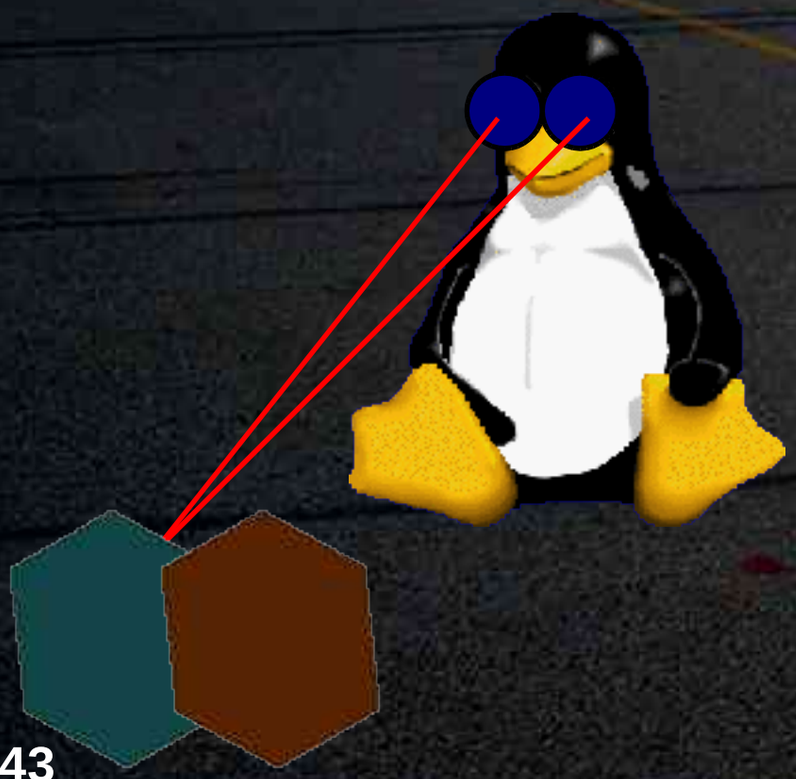
Tricks to using DC devices

- If you have trouble with the cameras, try reloading all the 1394 modules, this fixes most problems
- Gscanbus and coriander are nice debugging tools

- Important you configure your devices properly depending on the kernel and libDC version
  - Best advice is to use the newest 2.4 kernel
  - Use libDC version 9, and not 8 that you may be using
    - » mknod /dev/video1394/0 c 171 16
  - See the notes for other configurations

- Other devices such as /dev/raw1394 are typically ok
  - » mknod /dev/raw1394 c 171 0

Vision tracking



Intro to vision tracking

- We can use video cameras to capture the physical world
- Computers can analyse the images to extract information

- There has been a lot of talk about vision tracking coming soon but not really much action

- The ARToolKit is a nice example of a library we can use now to begin developing apps, without much knowledge

- I will show how ARToolKit works and some examples

ARToolKit processing

- Capture video frame
- Extract out edges
- Calculate rotation and translation
- Match against pattern database
- Profit! (well, a matrix)

- Walk through simpleTest

Pulling apart 4x4 matrices

- Computer graphics uses 4x4 matrices to represent translate, scale, rotate, and other linear operations
- Can be easily combined and handled in hardware
- ARToolKit computes its result as a 4x4 matrix
  - Uses [row][col] array notation

```
| a b c d |
| e f g h |
| i j k l |
| m n o p |
```

Example matrix operations

- Identity

```
| 1 0 0 0 |
| 0 1 0 0 |
| 0 0 1 0 |
| 0 0 0 1 |
```

- Trans

```
| 1 0 0 Tx |
| 0 1 0 Ty |
| 0 0 1 Tz |
| 0 0 0 1  |
```

- Scale

```
| Sx 0  0  0 |
| 0  Sy 0  0 |
| 0  0  Sz 0 |
| 0  0  0  1 |
```

- To extract out translation, grab
  - X = matrix [0][3]
  - Y = matrix [1][3]
  - Z = matrix [2][3]
- Rotation and scale are beyond the scope of this tutorial!

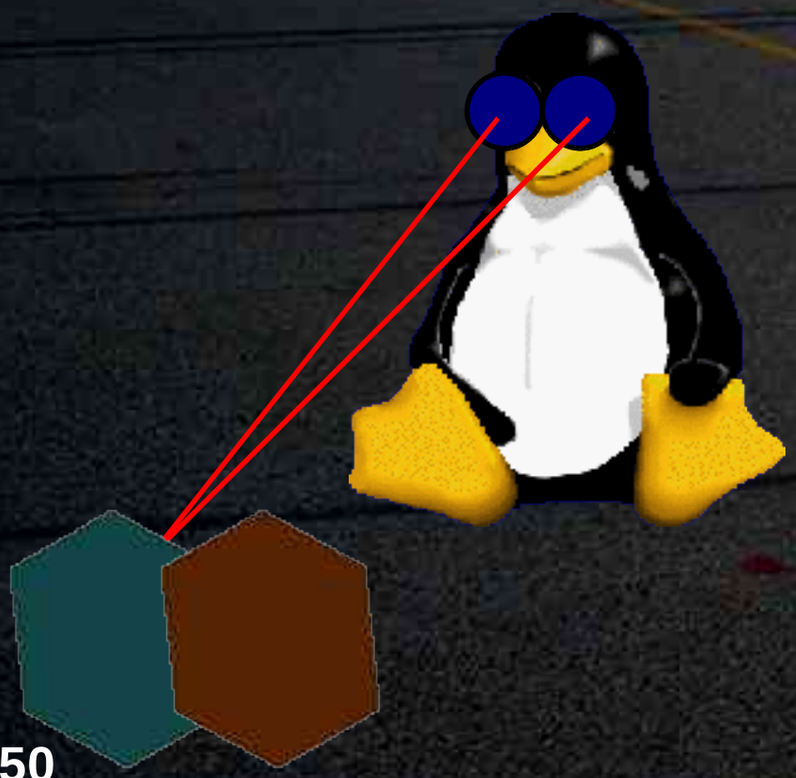
Using ARToolKit as a tracker

- I use ARToolKit as a tracker for my hands
  - Markers placed on each thumb
  - Extract out XYZ coordinates from the matrix
  - Project 3D coordinates to display to get 2D
- Use it to track your hands in front of your monitor?
  Attach a camera to a baseball cap that you wear?
- Generate real mouse events for X11?
- Control Gnome/ KDE?

Tracking position in a room

- Multiple cameras observe markers on the ceiling
- Inverse the matrix to find camera relative to the marker
- Must measure each marker relative to the room
- This one is a lot harder than it looks!

Custom hardware

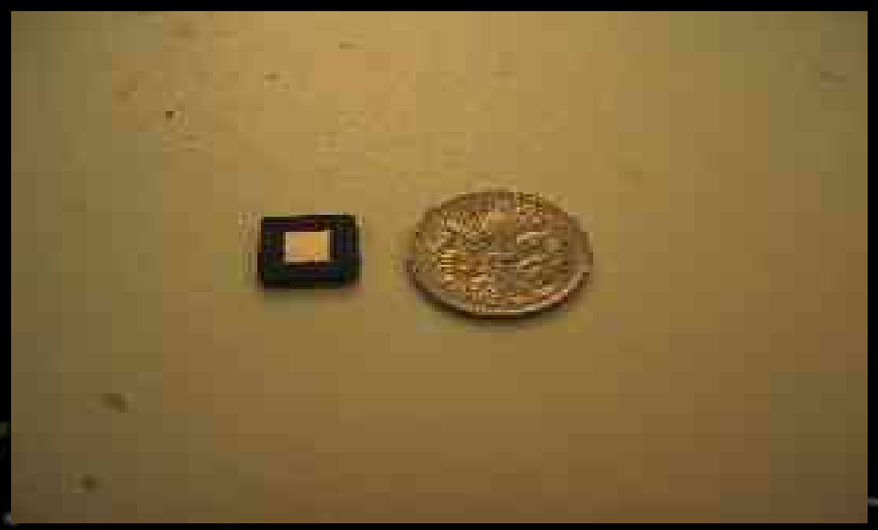


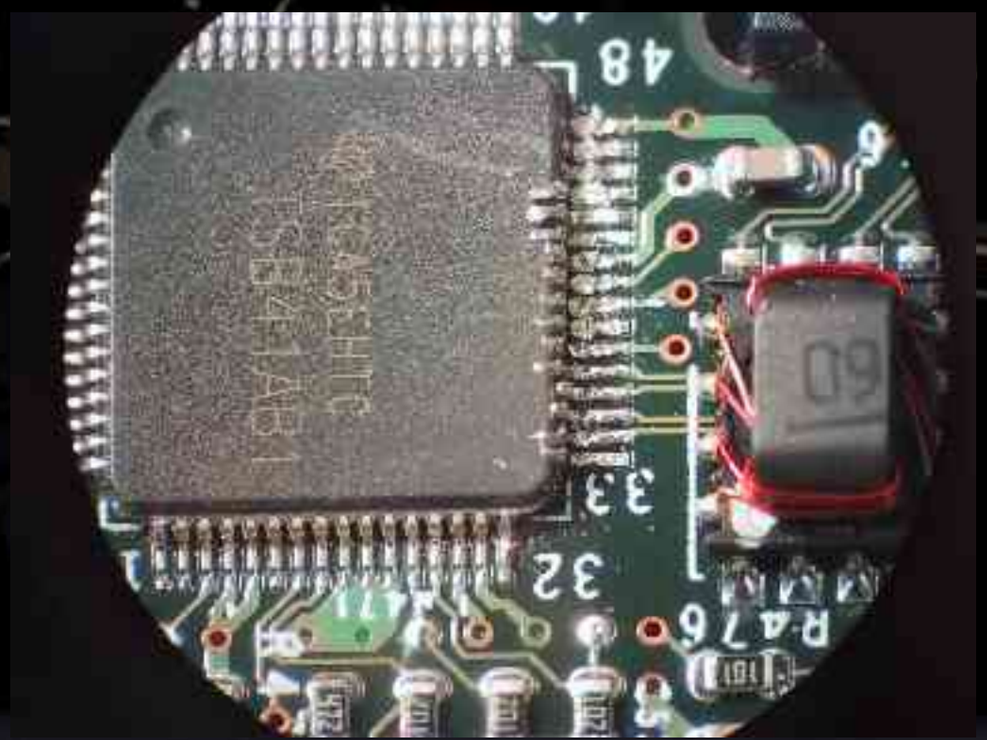
Important note

- Hacking is not just about software!
- It is about using tools to modify your hardware and also make new cool devices
  - Drills
  - Power Saws
  - Soldering Irons
  - Sticky Tape
  - Plastic and Metal
- And you make lots of mistakes along the way
- Combine the two for even more fun!

WCL Project Oxcart

- Repair damaged 1394 controller chip in Dell 8100 laptop
  - Texas Instruments TSB41AB1, 64-pin surface mount IC
  - 8 hour surgery
  - Saved $1200
  - Cost $5 plus time
  - Fun!





Interfacing technology

- Parallel port
- Serial port
- USB port

- PCs are becoming more complex and faster
- Interfaces are getting harder for hobbyists to play with
  - PCI, USB, Firewire are all very complicated
- New PCs are removing legacy ports
- Some nice interface chips to help out though

- The notes contains very detailed instructions which are glossed over here

Parallel port

- 25-pin D shell connector
- 8 data lines capable of +5V with low current
- Old ports are only single direction
- Other flow control lines also capable of data transfer
- Write directly to address using ioperm() and outb()
- LED CPU Meter

- CPU must bang out each byte manually
- Interrupt for each incoming byte or intensive polling

- Linux isn't really designed for any of these
- DOS is actually ideal for using these

Serial RS-232

- 9-pin D shell connector

- Baud rates up to 115,200 bps (slower than parallel)
- Much more friendly on the CPU with large UART buffers
- Simple cables with only 3 wires needed
- Requires a port for each device, limited on laptops
- Open up device and use standard I/O calls on an FD

Serial RS-232

- Needs extra hardware at the remote end
  - Basic Stamp II or other microcontroller
  - Performs intensive I/O tasks without affecting CPU
- There are only a small number of serial ports
  - Use USB interfaces - kernel maps to standard /dev
  - FTDI FT8U232AM chip
  - Keyspan series converters
  - Don’t bother with PCMCIA, not enough slots

Dumb controller boxes

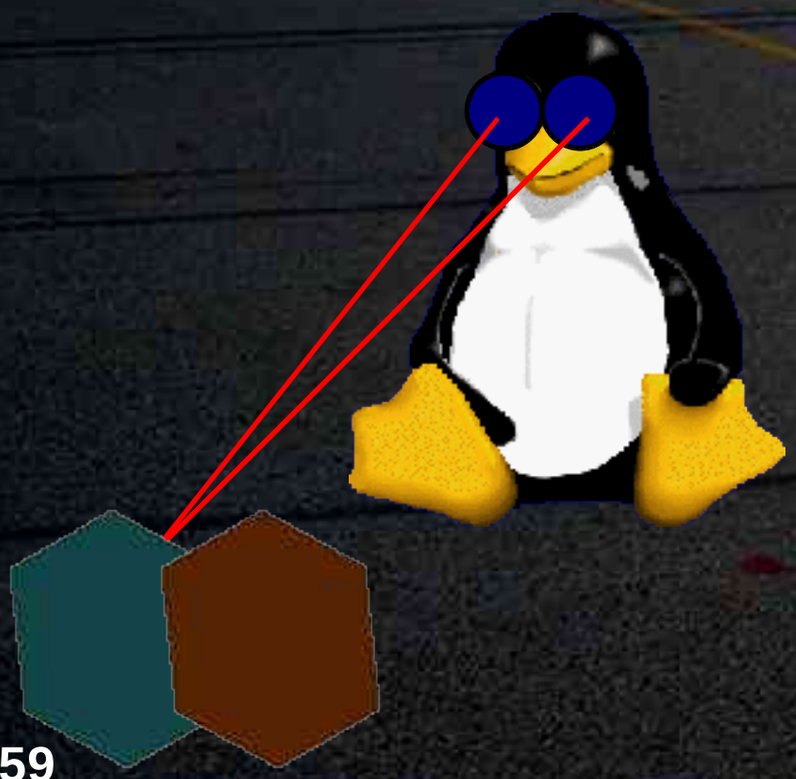
- You can buy boxes that have a number of input and output pins connected via USB, serial, parallel
- These devices have no smarts and must be controlled continuously and will use up a lot of CPU time
- Try to use a microcontroller if you can
- MCU provides real time functionality and only makes the CPU deal with it when something interesting happens
- I used this in my glove controller very successfully

Cheap hacks

- USB mice have three or more buttons
- Interfaces are already built
- Simply cut open and solder custom switches
- Applications that use a mouse need no modifications
- Use an old mouse or get a cheap one

- Why work when you don’t have to?
- Hacking existing gear is much easier and saves time and money

Conclusion



Conclusion

- We have talked about a lot of material today
- We had to gloss over a lot because of time restrictions
  - The notes contain lots of detail about everything I have talked about today
  - Code examples are on the conference CD
- Talked about video capture and display, 3D vision tracking libraries, and building custom hardware
-
- I look forward to seeing what people have built by the next Linux Conf!

Questions?

- Good luck, and don't fry your hardware!
-
- Wayne Piekarski
- University of South Australia
- Wearable Computer Lab

- http://www.tinmith.net
- http://wearables.unisa.edu.au
- wayne@cs.unisa.edu.au