Object Recognition in Video Games

Tony Morelli

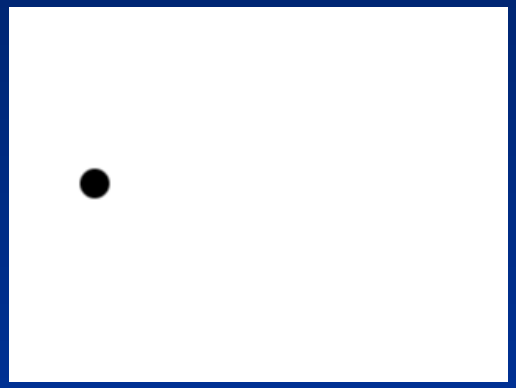
CS674 Digital Image Processing

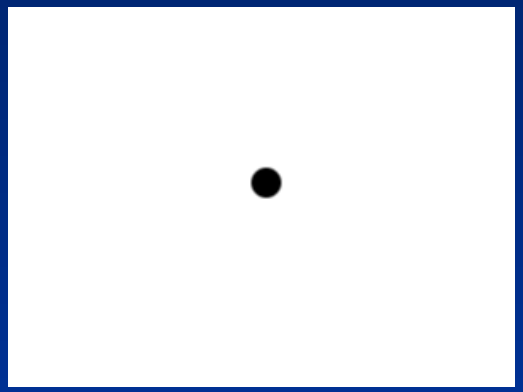
Project Description

- Can my character be differentiated from enemies?
- Can this be done in real time?

Detecting Motion

- Take 2 Frames and Subtract them.
- Result is the difference/motion

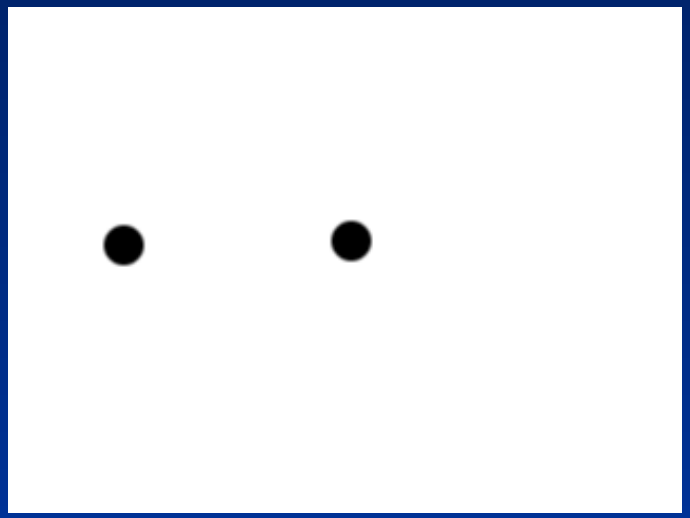




- Frame 1 Frame 2

Detecting Motion

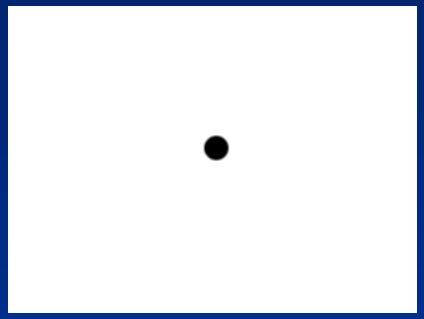
- Difference



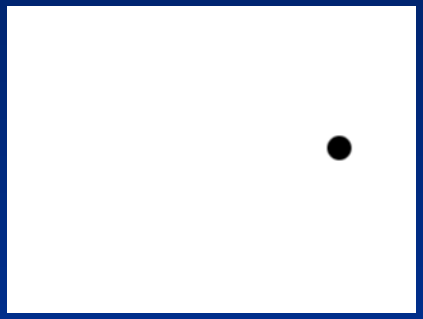
- Which direction is the shape moving?

Detecting Motion

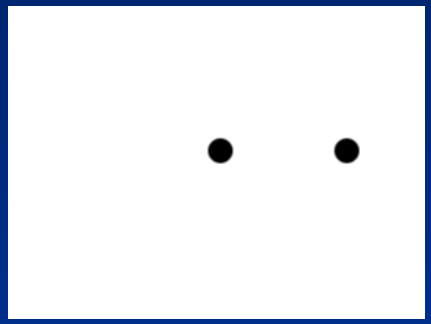
- To find the direction of change, look 1 frame forward.



Frame2



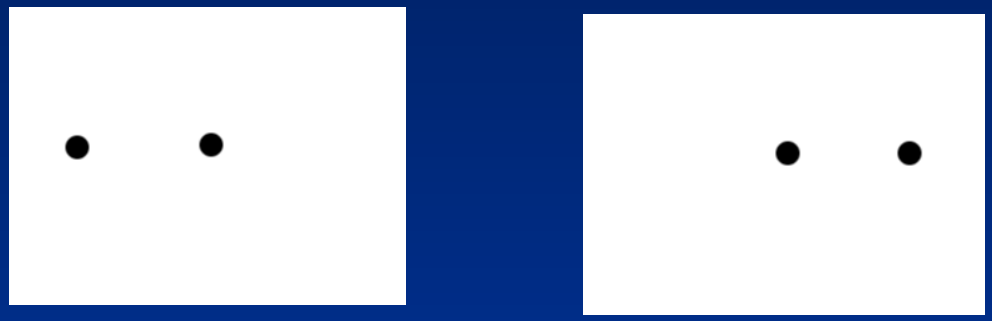
Frame 3



Diff/Motion

Detecting Motion

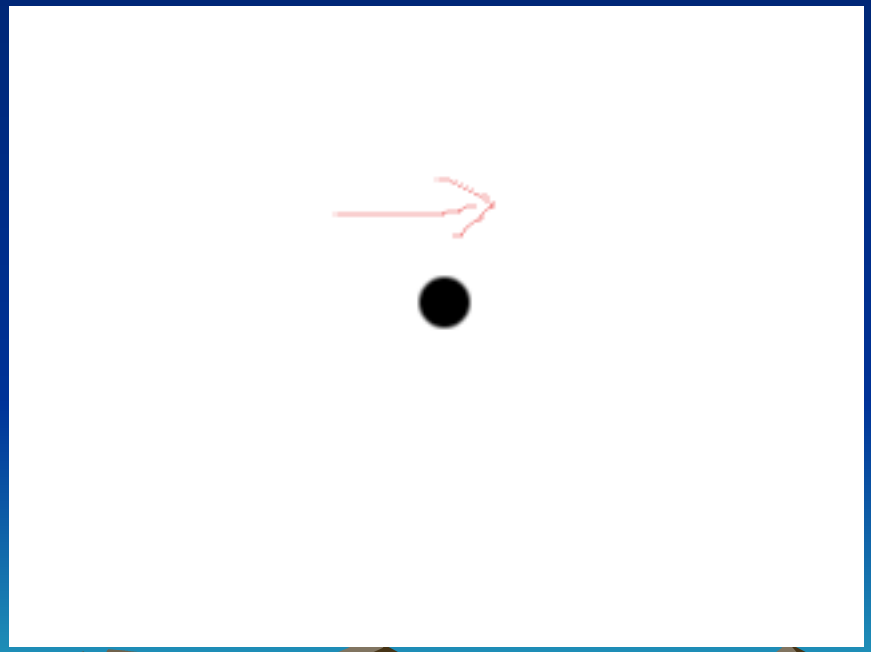
- Compare Difference from 1-2 and 2-3



- Since the regions are the same size, the location of the region that is the same between the 2 pictures is the current location

Detecting Motion

- Examining where the object is for the 3 frames tells you direction



Detecting Motion

- Try Concept with a video game
- Used Super Mario Bros – Simple 2d game
- Game was played in emulator, screenshots were taken as game was played

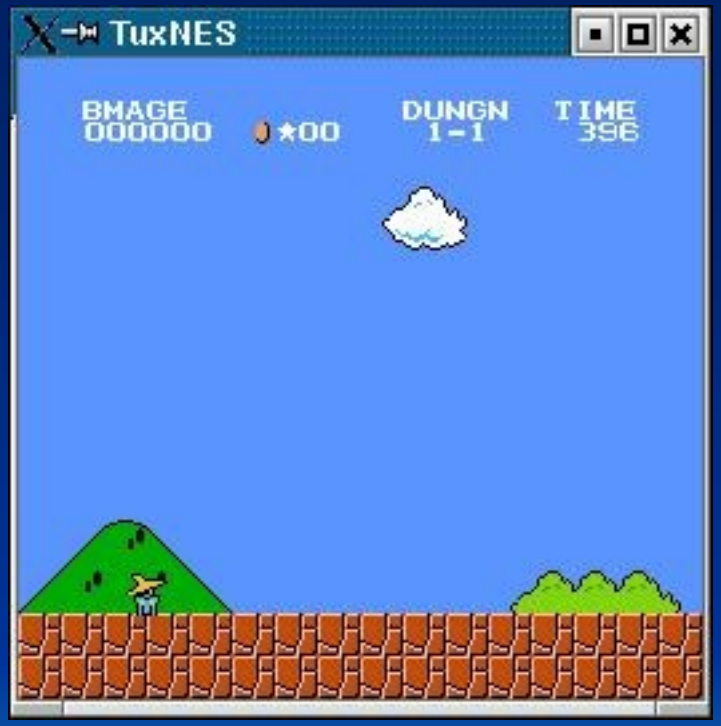
Detecting Motion

- Frames were stored as P6 PGM
- 3 Bytes per pixel
- Resolution 320x240

Detecting Motion

- Program was written in C++
- Images read in from disk and stored in memory
- Pixels were subtracted as in example

Detecting Motion



Frame 1

Frame 2

Detecting Motion

Differences

Regions with motion

Detecting Motion

- As shown in the previous slides, differences are taken between First/Second and Second/Third
- This results in motion and direction for all regions
- Since I was pushing right, any region moving left is considered an enemy, everything else is what I am controlling

Detecting Motion

- Calculation of object motion took a little over 1 second.
- Region recognition was easy as the colors were perfect (coming directly from emulator)
- Program had to be told what buttons I was pressing to determine which object I was controlling and which were enemies

Real Time Motion Detecting

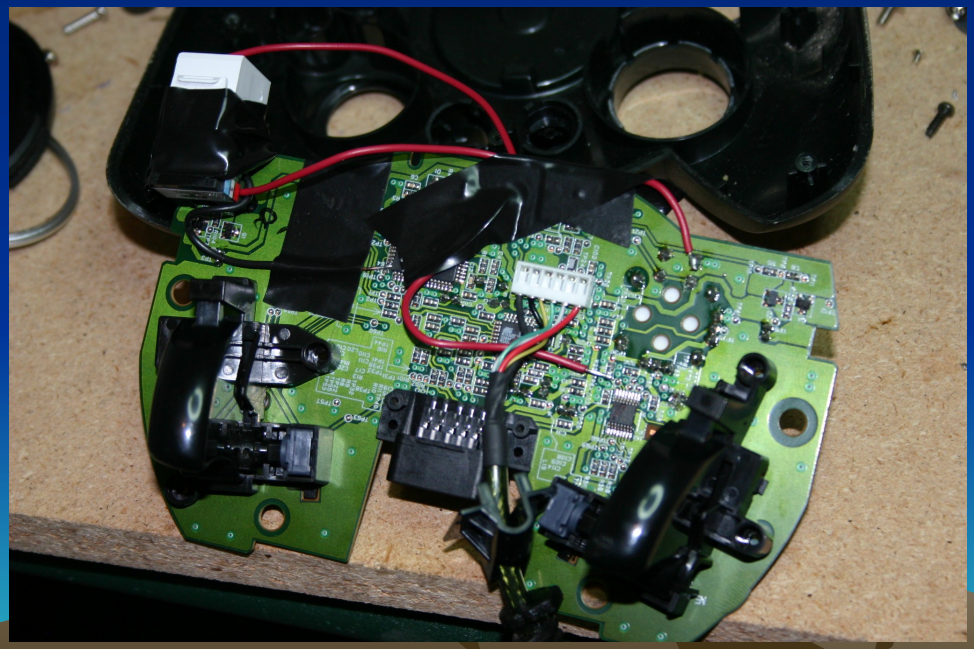
- Instead of reading from disk, get images from web cam directly from memory
- Examine a more ‘real world’ example by using web cam instead of perfect emulator image
- Have application control the direction, so it knows which direction to expect itself and enemies to be moving

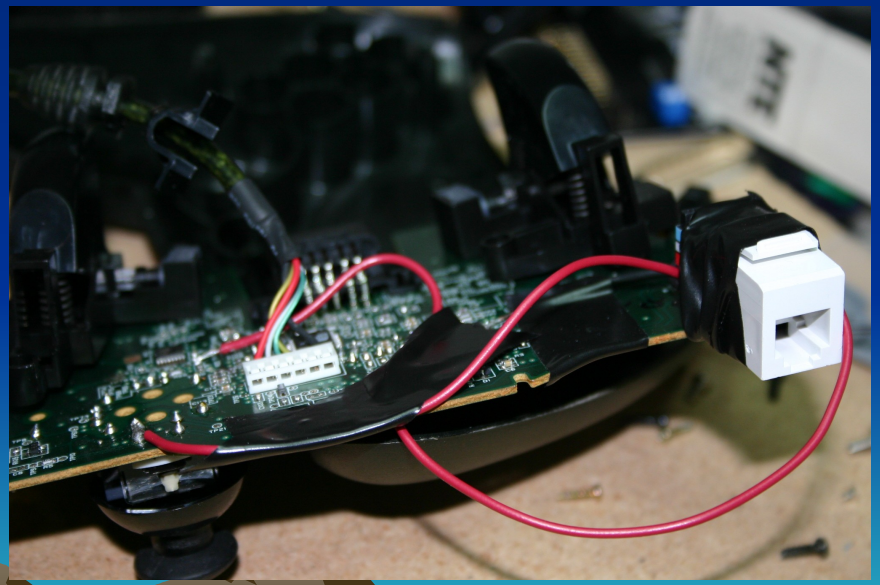
Real Time Motion Detecting

- ReWritten in VB.NET to better interface with web cam

Real Time Motion Detecting

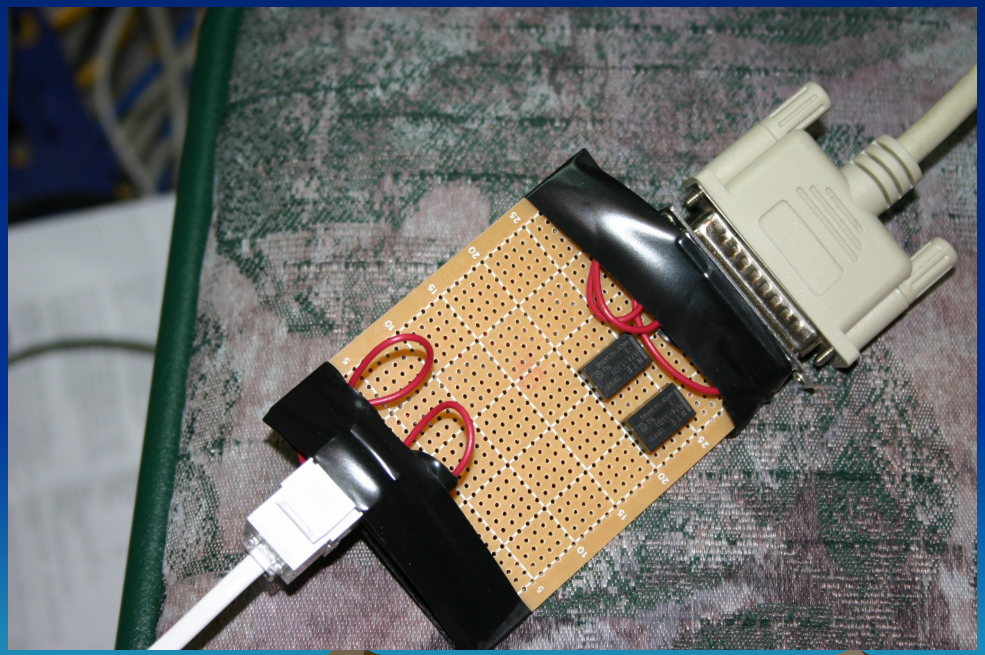
- Emulation moved to run on XBOX
- Computer will now control the emulator
- Modified XBOX Controller

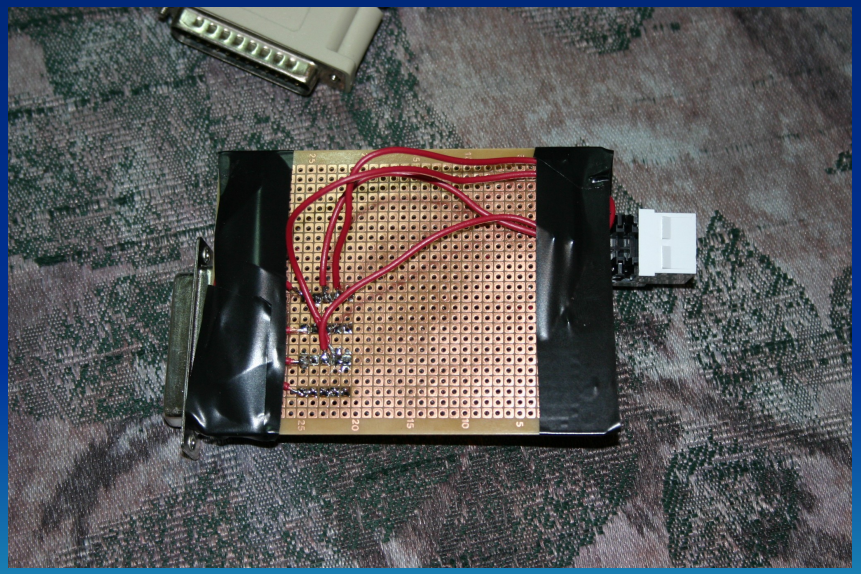




Real Time Motion Detecting

- Computer controls XBOX controller via parallel port





More Image Processing

- More details need to be kept about objects
  - Size, Color
  - Not all objects approaching me are bad
- Score needs to be recognized