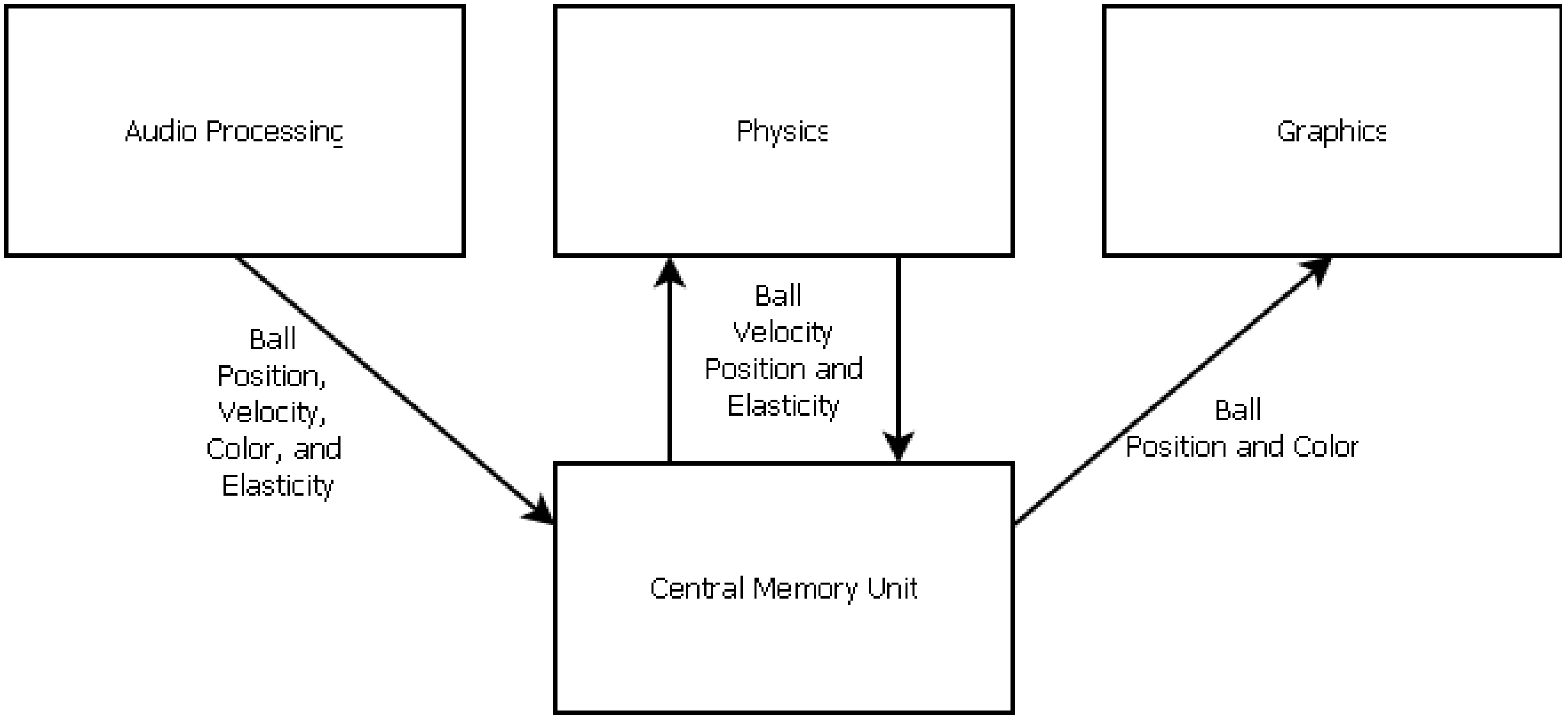
Bellagio Fountain Simulation

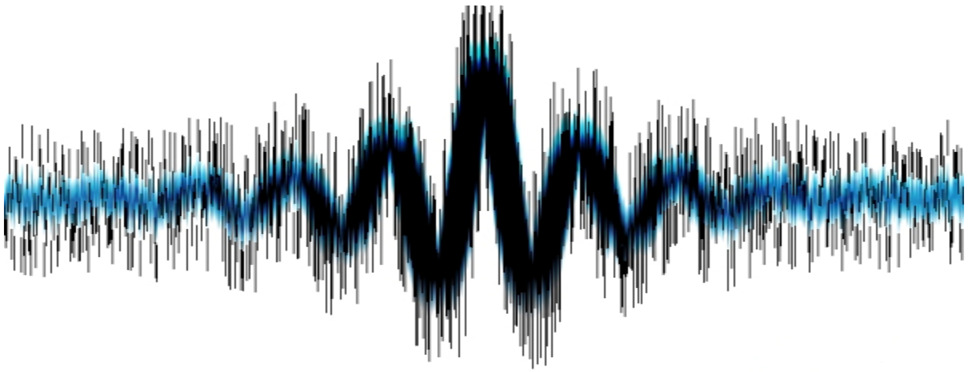
George Rossick • Allen Yin • Joseph Lane

PROJECT DIAGRAM

Objective: Create an entertaining simulation of the Las Vegas Bellagio fountains including real-time audio characterization, realistic physical interactions, and a 3 dimensional graphical environment.



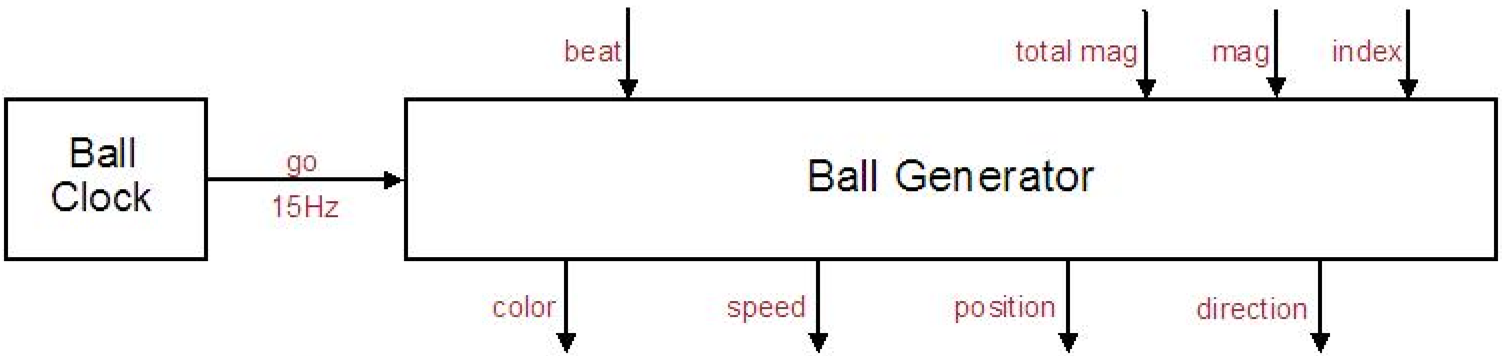
Audio Processing



- Objective
  - Generate fountain behavior and appearance based on audio features.
- Audio features
  - Amplitude
  - Frequency
  - Beat
  - Overall Energy
- Fountain Characteristics
  - Ball color
  - Speed and direction
  - Position

Ball Generator

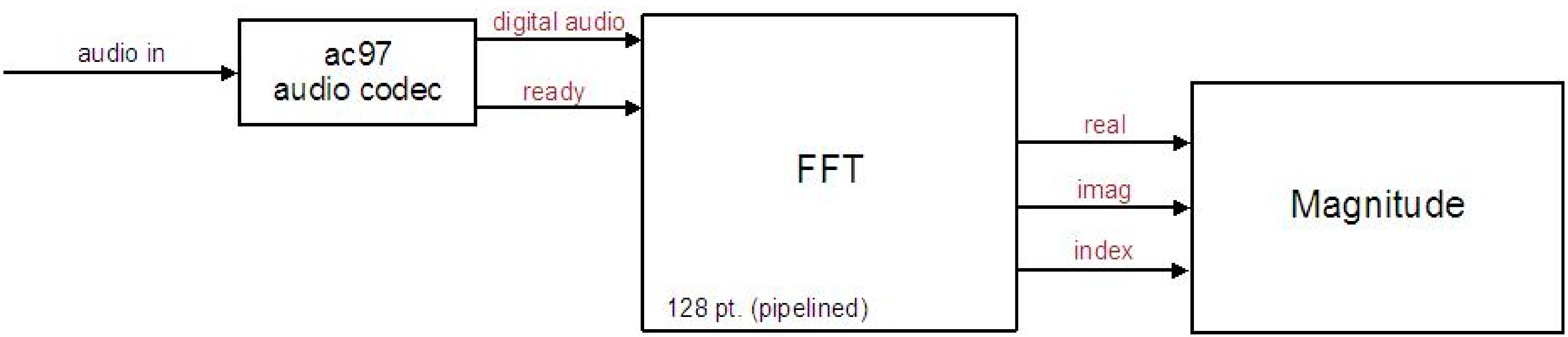
- Converts magnitude, frequency, beat and total energy to fountain behavior
  - Magnitude → Ball speed
  - Frequency → Fountain
  - Beat → Ball color (songs with beat)
  - Total energy → Ball color (songs without beat)



*27Mhz clock and reset signal to all modules

Amplitude and Frequency

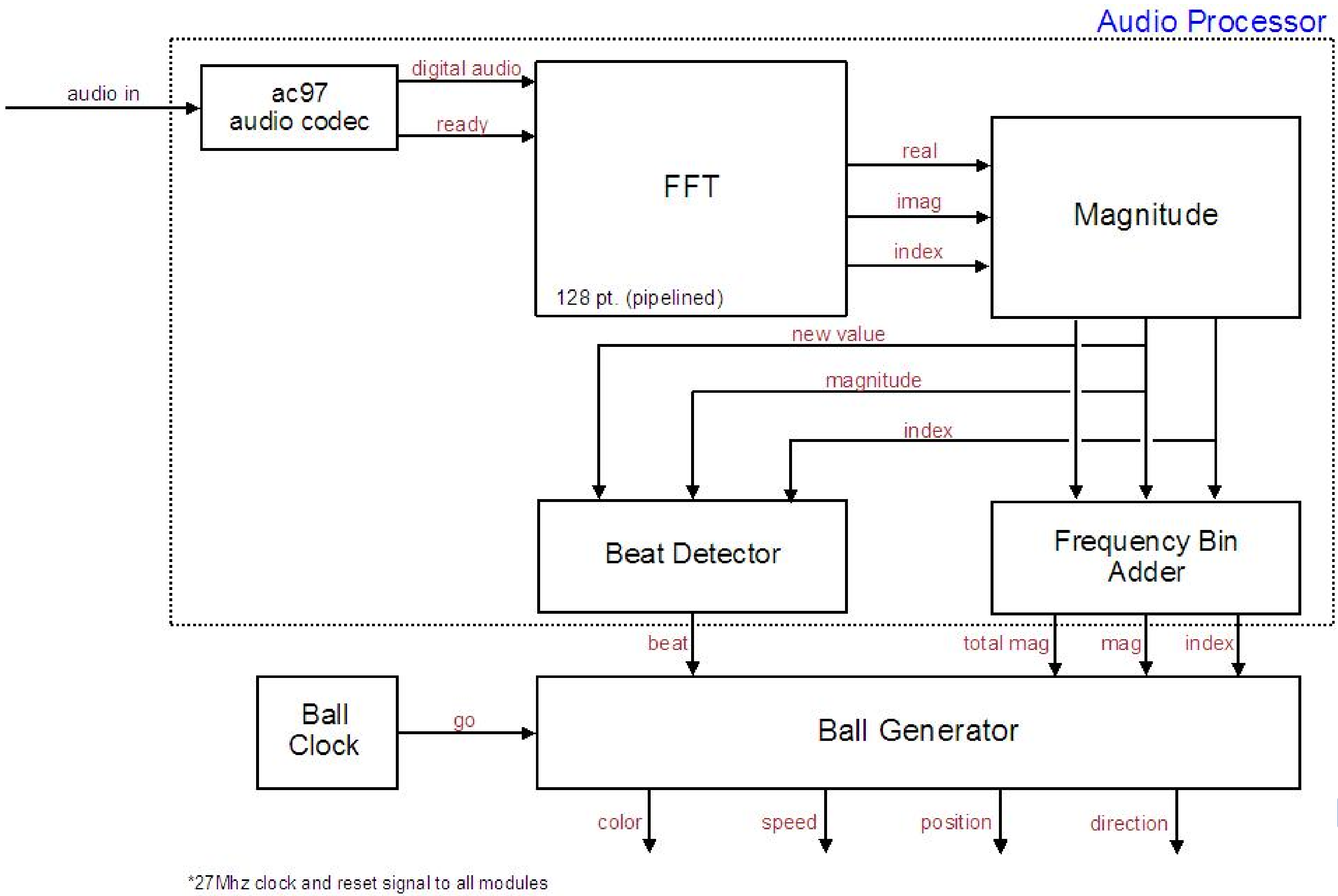
- 128 point FFT: time domain→ frequency domain
- Magnitude module converts real/imaginary values to signal magnitude



Beat and Total Energy

- Beat detection
  - Time domain – simple to implement, great for instrumental. Sensitive to noise and vocal energy
  - Frequency domain – more complicated, robust to vocals and noise.
  - Low freq, < 350Hz • High freq, > 1000Hz
  - Instantaneous energy > C*average energy ⟶ beat
  - Use rolling averages over 46 samples of 1024 values
- Total Energy
  - For songs without clear beats: classical, a cappella
  - Sum over all magnitudes

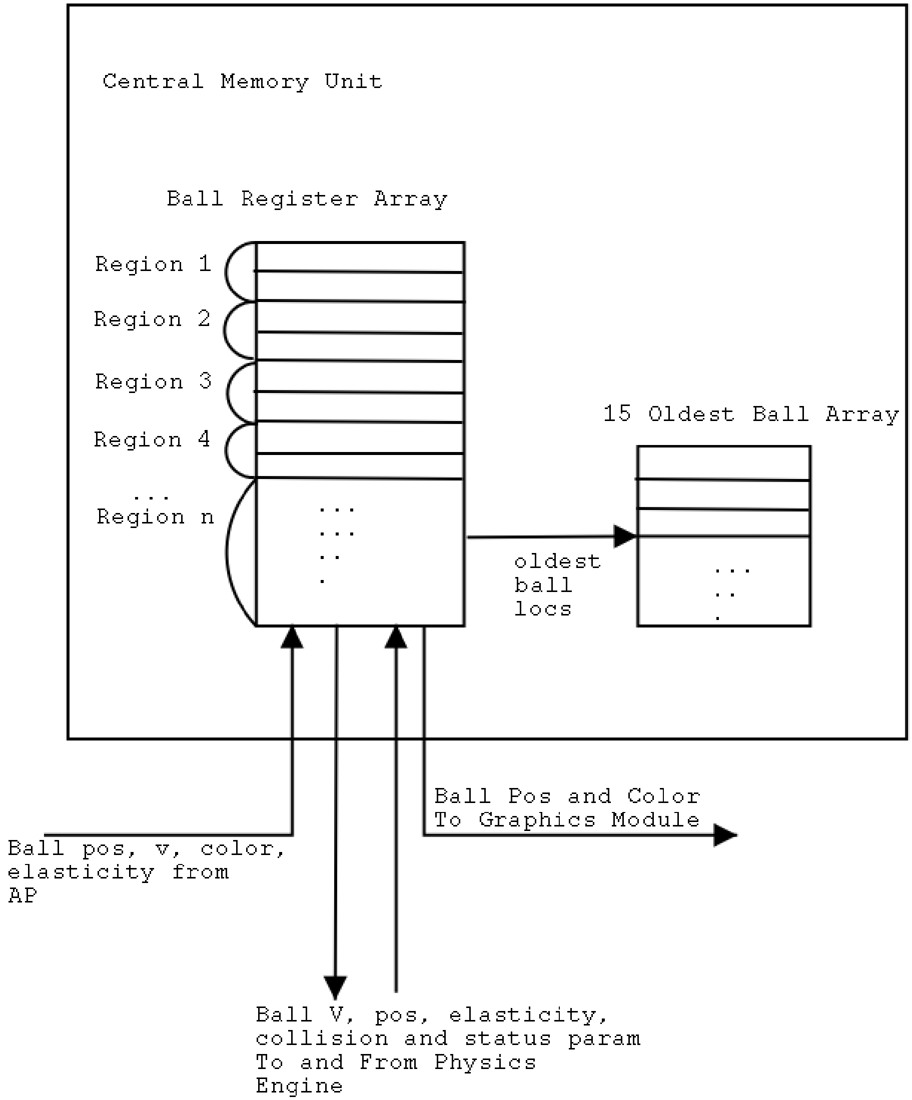
Audio Processor Diagram



Physics Engine

- Model the physical interactions between balls
  - Gravity
  - Collision Detection
  - Collision physics calculations

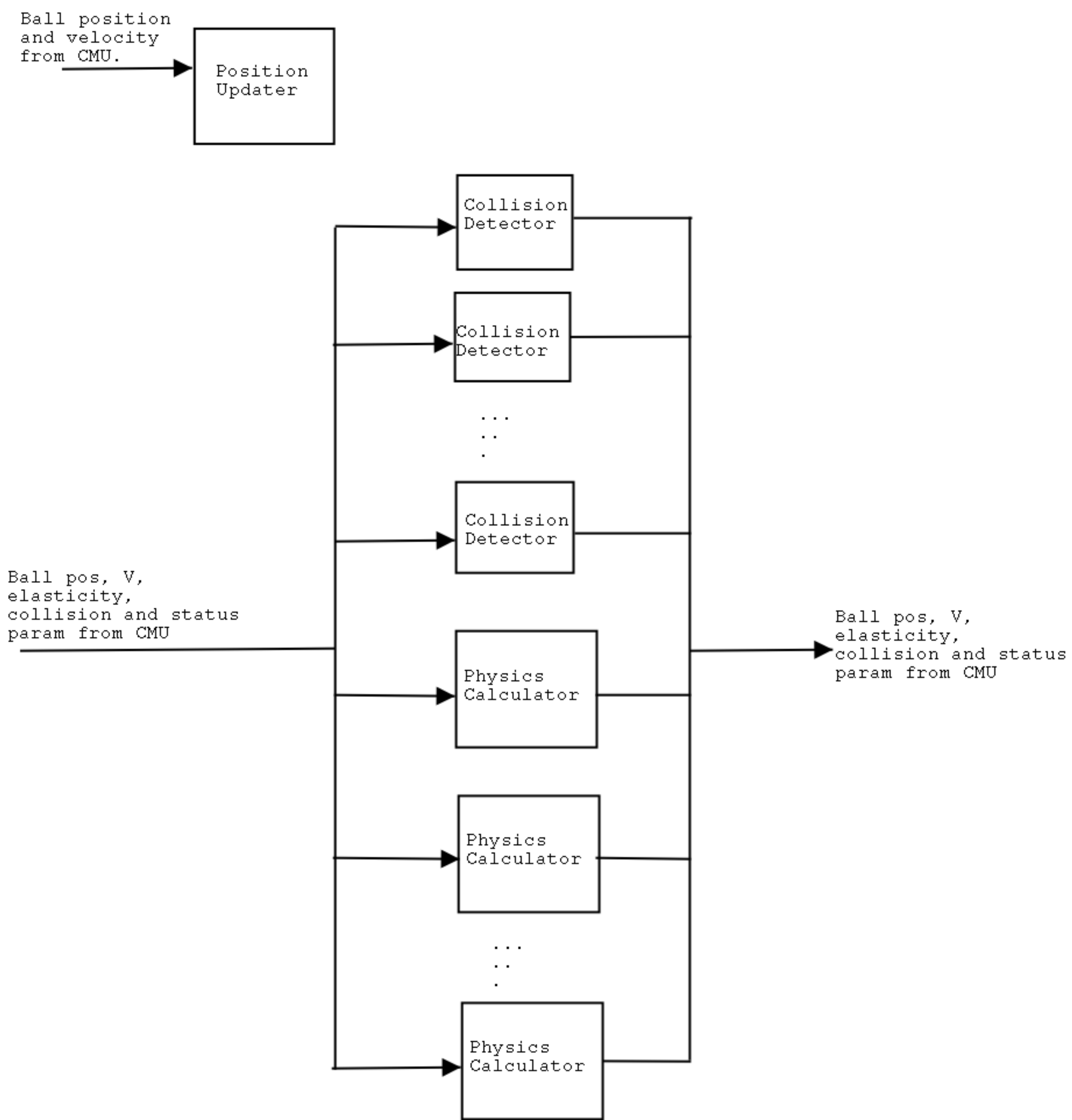
Central Memory Unit as Data Structure for Balls



CMU as Data Structure for Balls

- 1,000 registers represent 1,000 balls
- Each register holds
  - Coordinates
  - Velocity
  - Elasticity Coefficient
  - Time
  - Collision Parameter
  - Status Parameter
- Organization of balls into spacial regions for efficient collision detection
- 15 oldest balls array makes refill easier

Collision Detection and Physics Calculations



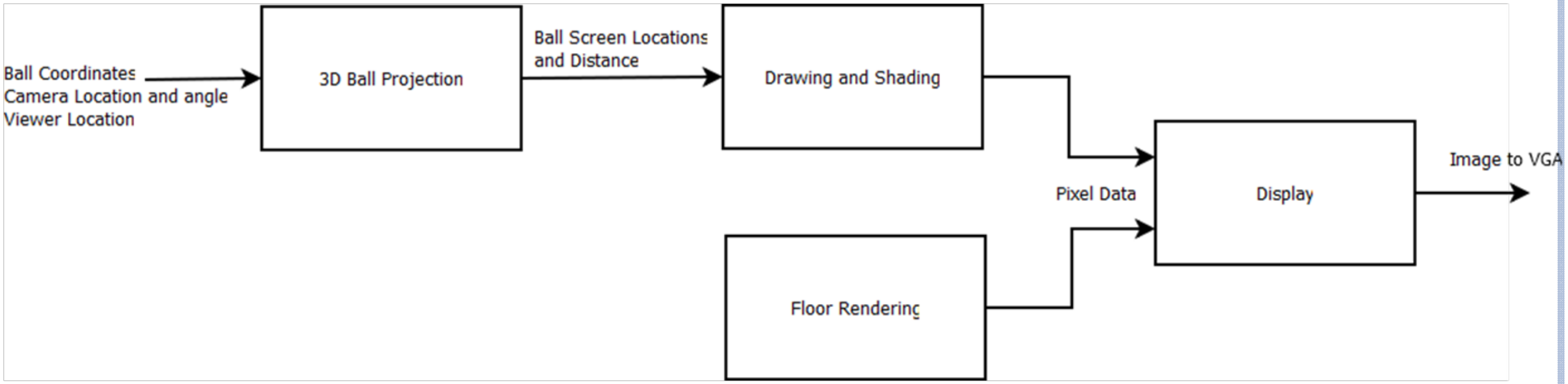
Collision Detection

- After velocity update (including gravity and wall bounce), iterate over each region to find collision between pairs
- Multiple collision detectors allow for parallelism
- Metrics based on distance
- 3-D space required parallel vector detections
- Update the collision parameter accordingly

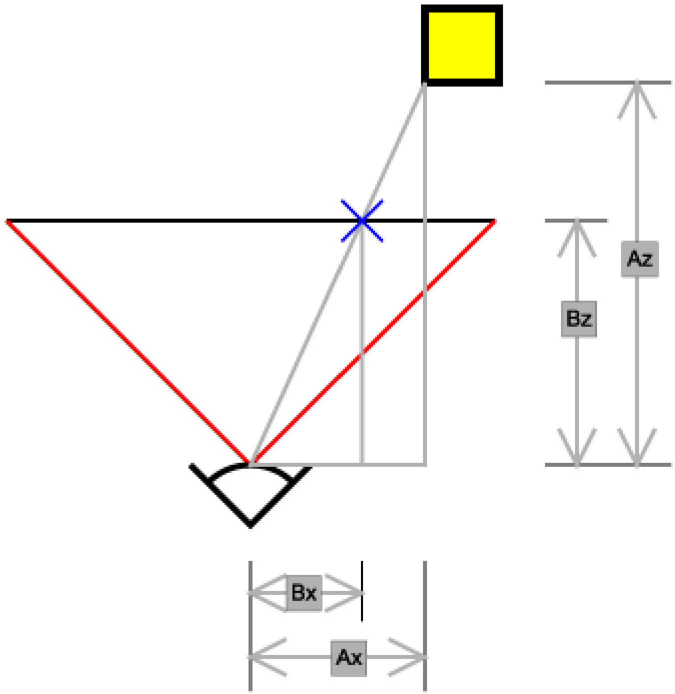
Physics Calculations

- In parallel with collision detections
- Post-collision velocity calculations by converting into center of mass coordinates and back
- Only use physics calculators pairs involved in collisions
- Multiple physics calculators allow for parallelism.

Graphics Module



3D Projection onto a 2D Plane



- 4 Variables:
  - Location that the center of the ball is at
  - Location of the camera (center of the screen)
  - Angle of the camera
  - Location of the viewer
- Determines location of the center of a ball on the screen

Image from http://en.wikipedia.org/wiki/3D_projection

Drawing the Floor Onto the Screen

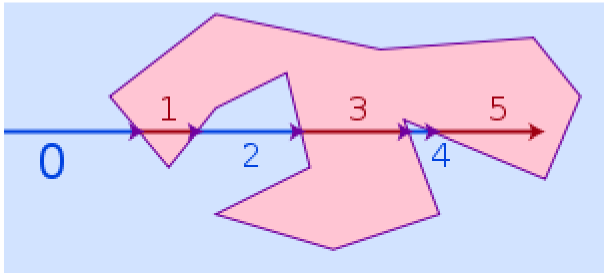
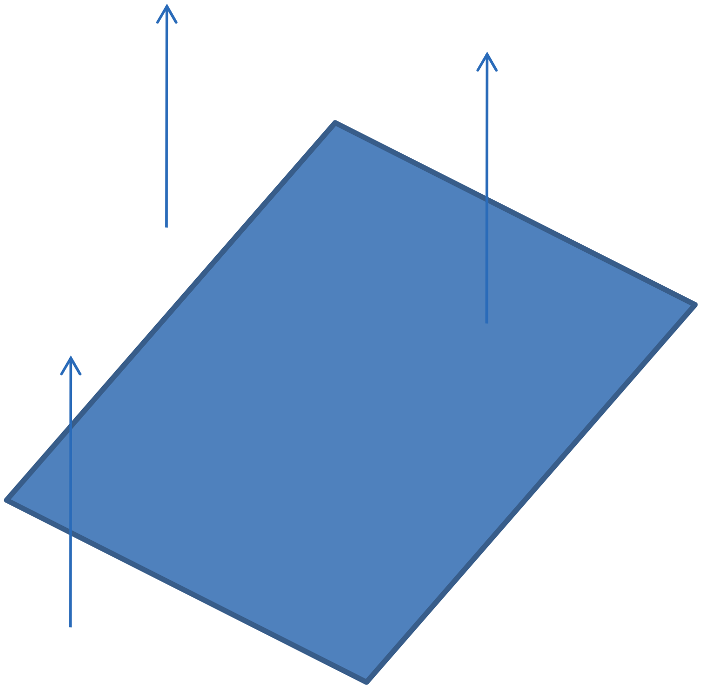


Image From http://en.wikipedia.org/wiki/Point_in_polygon



- Project the four corners of the floor onto the screen
- For each point on the screen draw a ray
- If the ray passes through an odd number of boundaries it is inside the floor otherwise it is outside.

Drawing and Shading

Drawing

- Once we know the X and Y position of each ball on the screen and the distance of the ball from the screen we can draw it
- Z-Buffer: array of the depth of each pixel
- If the old value in the Z buffer for a pixel is greater than the new value the pixel is overwritten

Shading

- Conventional shading methods are too expensive for the number of balls on screen
- Shading done in the YUV color space
- Pre-rendered Y value
- UV determined by audio processing

Timeline

Week	Graphics	Physics	Audio
11/16	Particle graphics	2-D model without sorting	128 pt. FFT with Mag.
11/23	Balls	2-D model with sorting	Beat Detection
11/30	Shading	Expand to 3-D and parallelize	Fountain behavior
12/7	Debugging	Debugging	Debugging