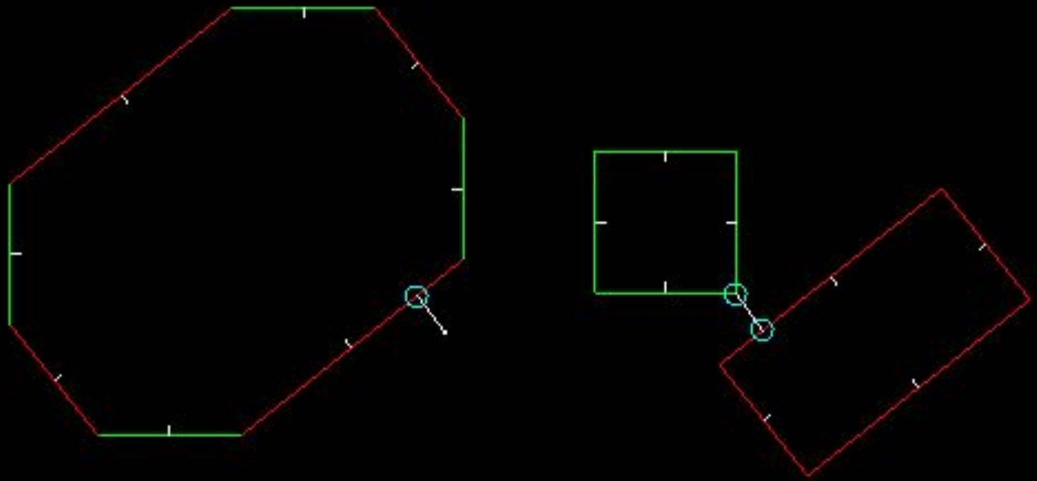
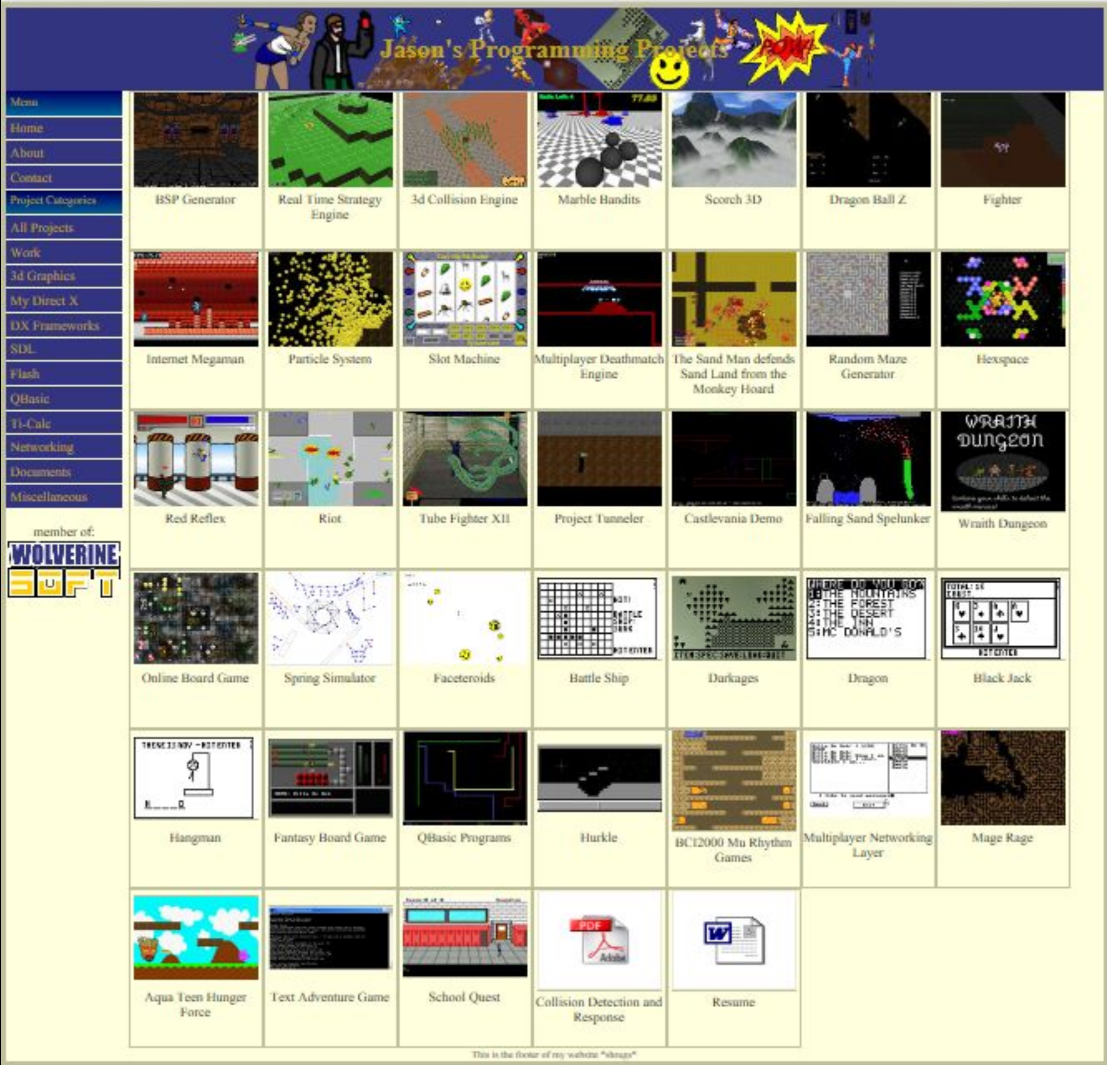
Game Geometry and Physics

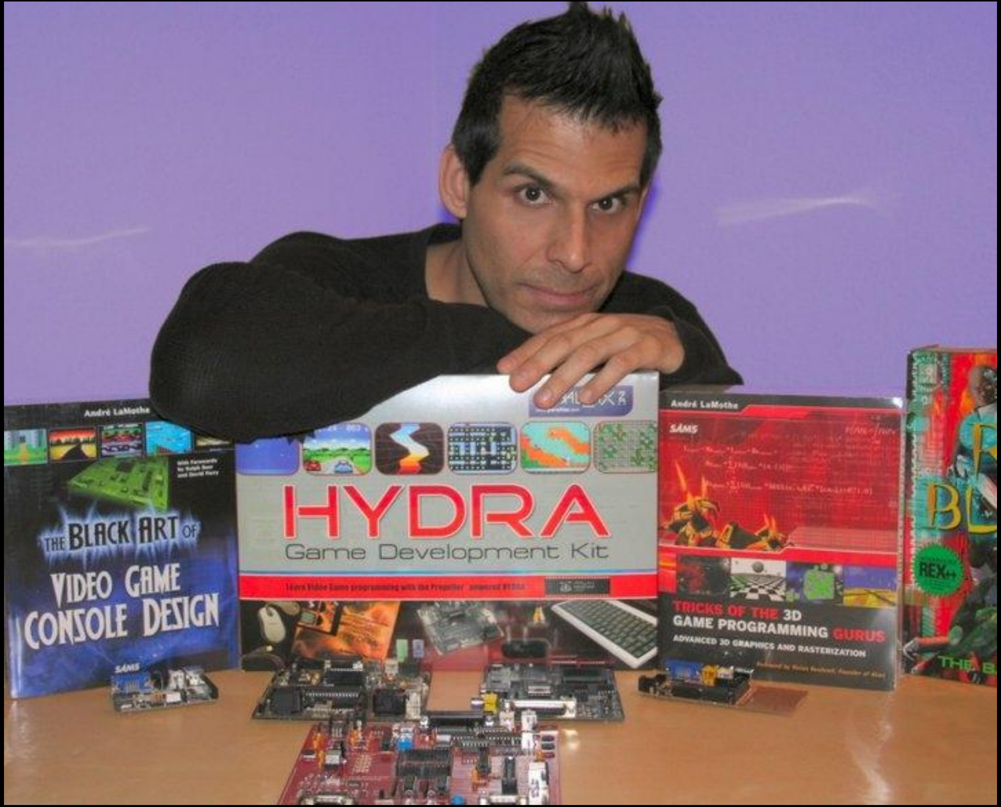
rolling out collision and physics code

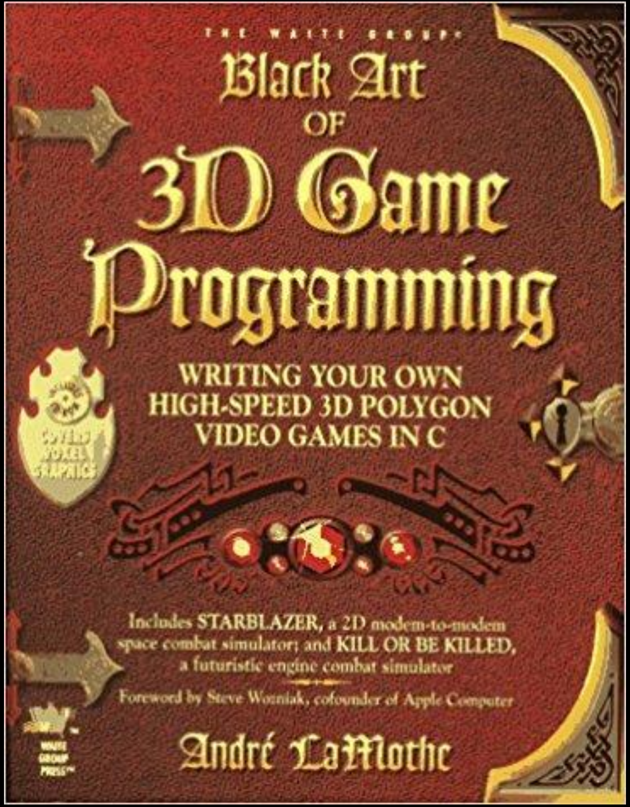


jeaton.matero.net



André LaMothe


André LaMothe
THE BLACK ART OF VIDEO GAME CONSOLE DESIGN
SAMS
HYDRA
Game Development Kit
André LaMothe
SAMS
TRICKS OF THE 3D GAME PROGRAMMING GURUS
ADVANCED 3D GRAPHICS AND RASTERIZATION
REX++


THE WAITE GROUP®

Black Art OF 3D Game Programming

WRITING YOUR OWN HIGH-SPEED 3D POLYGON VIDEO GAMES IN C

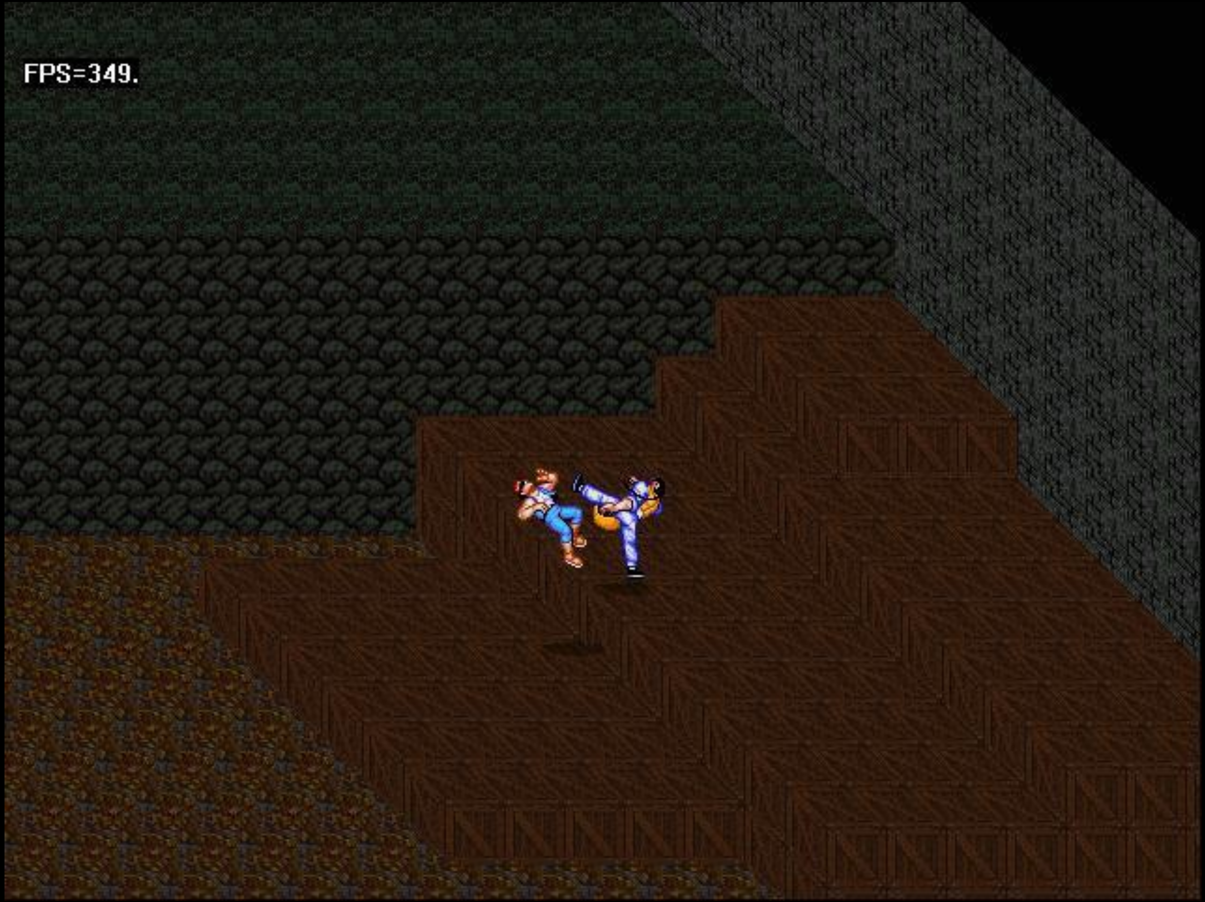
COVERS VOXEL GRAPHICS

Includes STARBLAZER, a 2D modem-to-modem space combat simulator; and KILL OR BE KILLED, a futuristic engine combat simulator

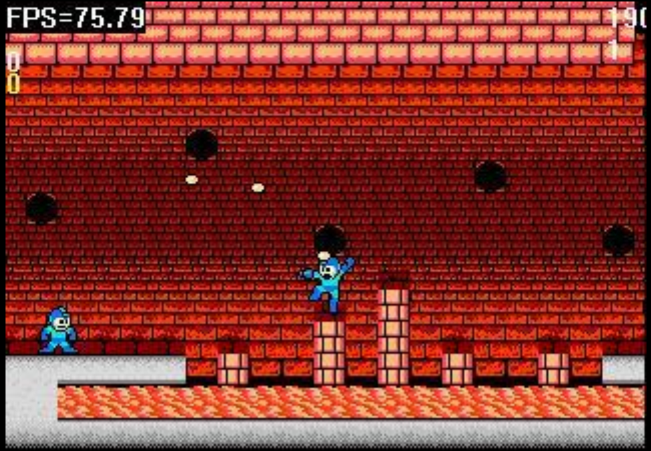
Foreword by Steve Wozniak, cofounder of Apple Computer

WAITE GROUP PRESS™

André LaMothe

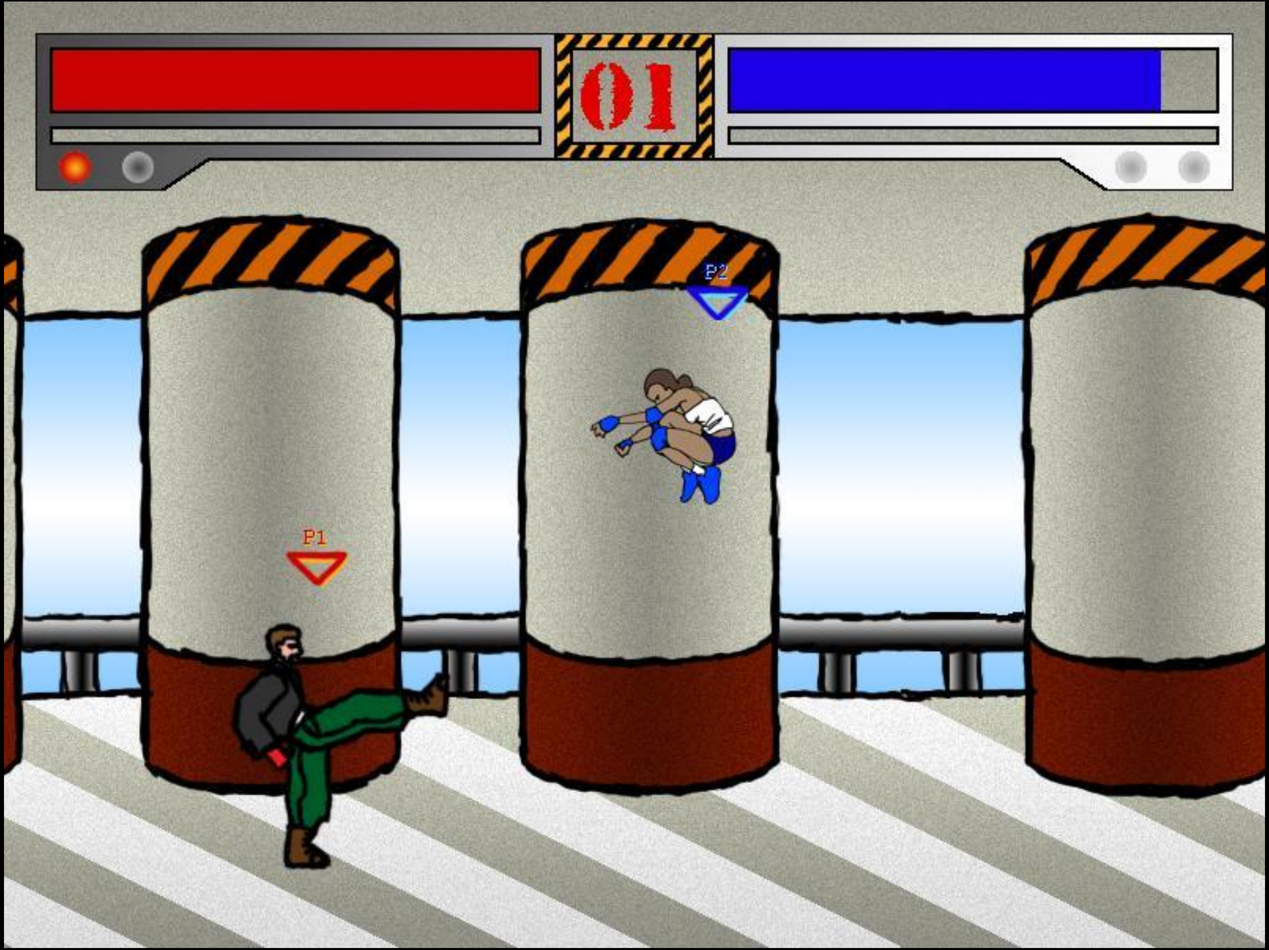


Very first demo made in high school, uses intricate pseudo 2.5d engine with “planes” and complete with fight game hit box editor.

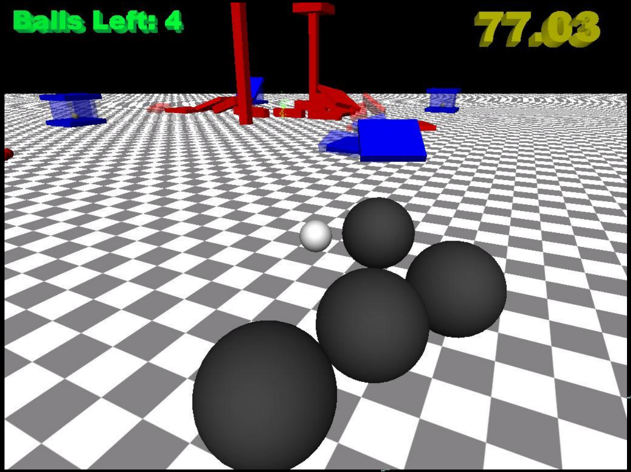


First attempt at networking code, first year in college

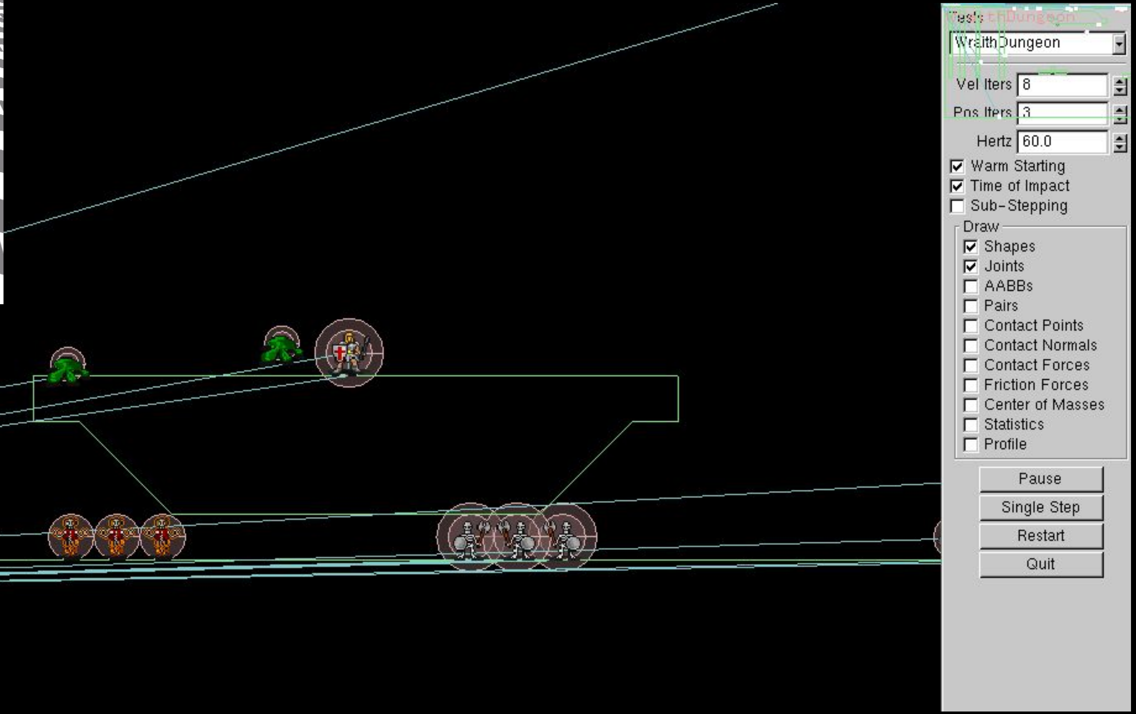
Uses my first demo’s code without the 3rd dimension



Open Dynamics Engine



Box2d



Tests

WraithDungeon

Vel Iters 8

Pos Iters 3

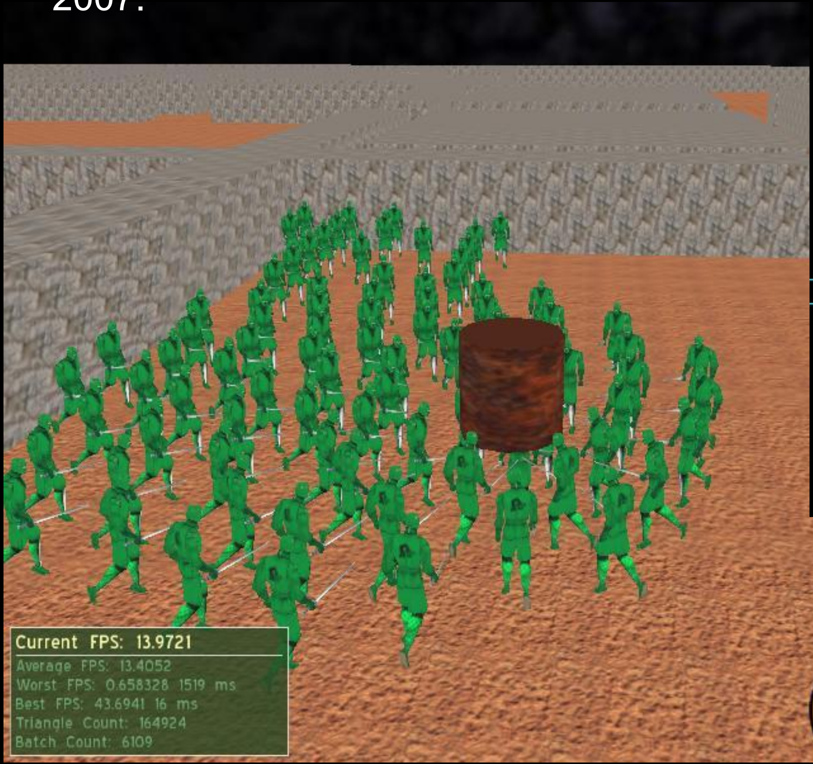
Hertz 60.0

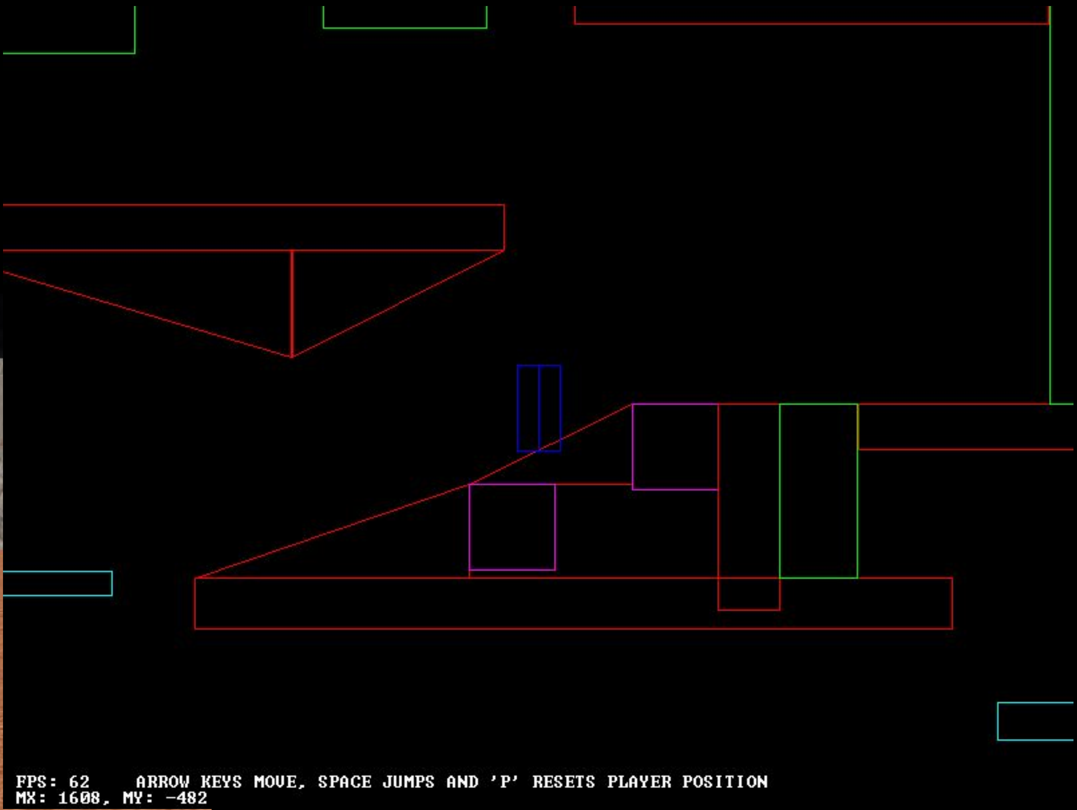
☑ Warm Starting
☑ Time of Impact
☐ Sub-Stepping

Draw

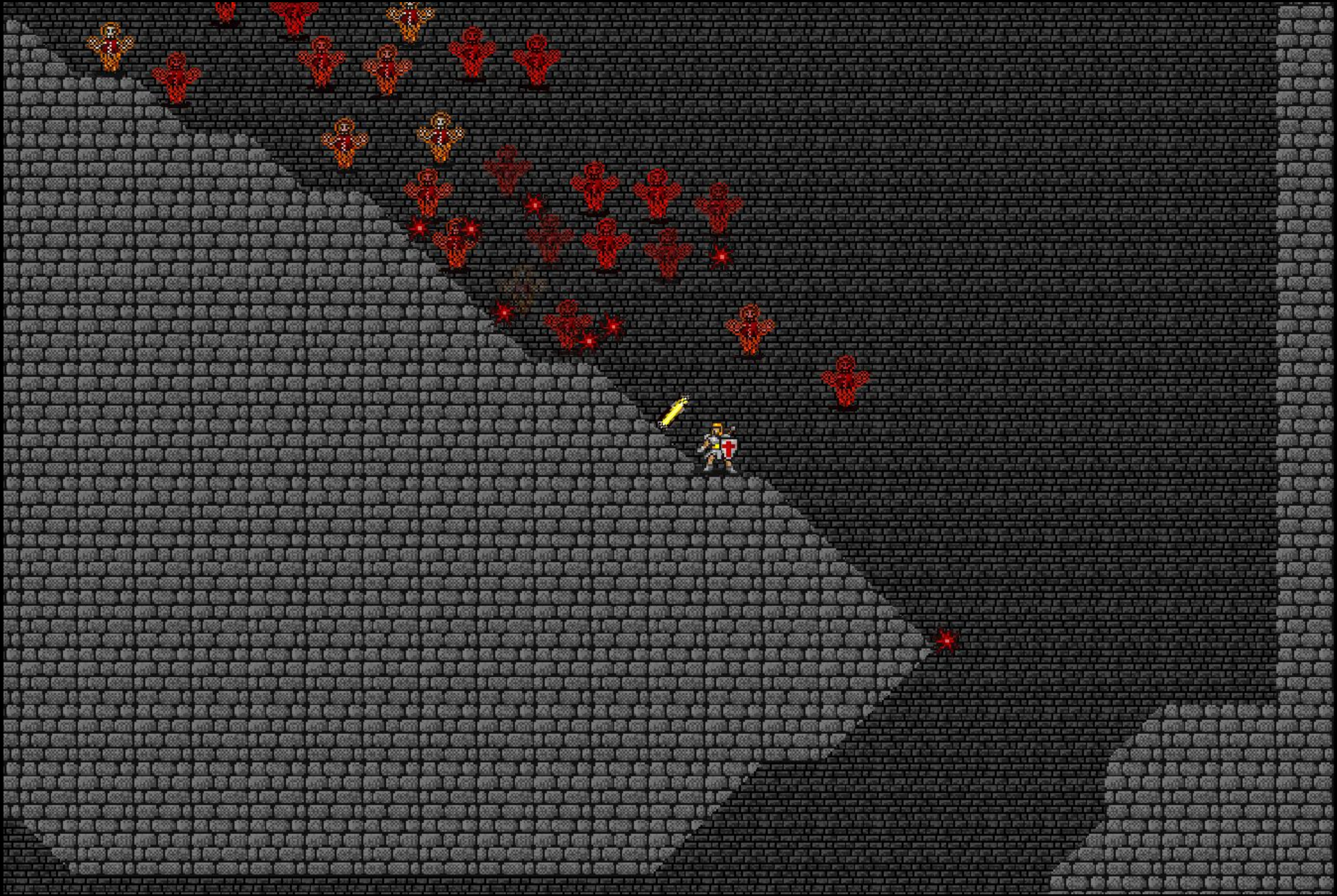
☑ Shapes
☑ Joints
☐ AABBs
☐ Pairs
☐ Contact Points
☐ Contact Normals
☐ Contact Forces
☐ Friction Forces
☐ Center of Masses
☐ Statistics
☐ Profile

I was developing this 3d collision engine to work with all blocks back in 2007.

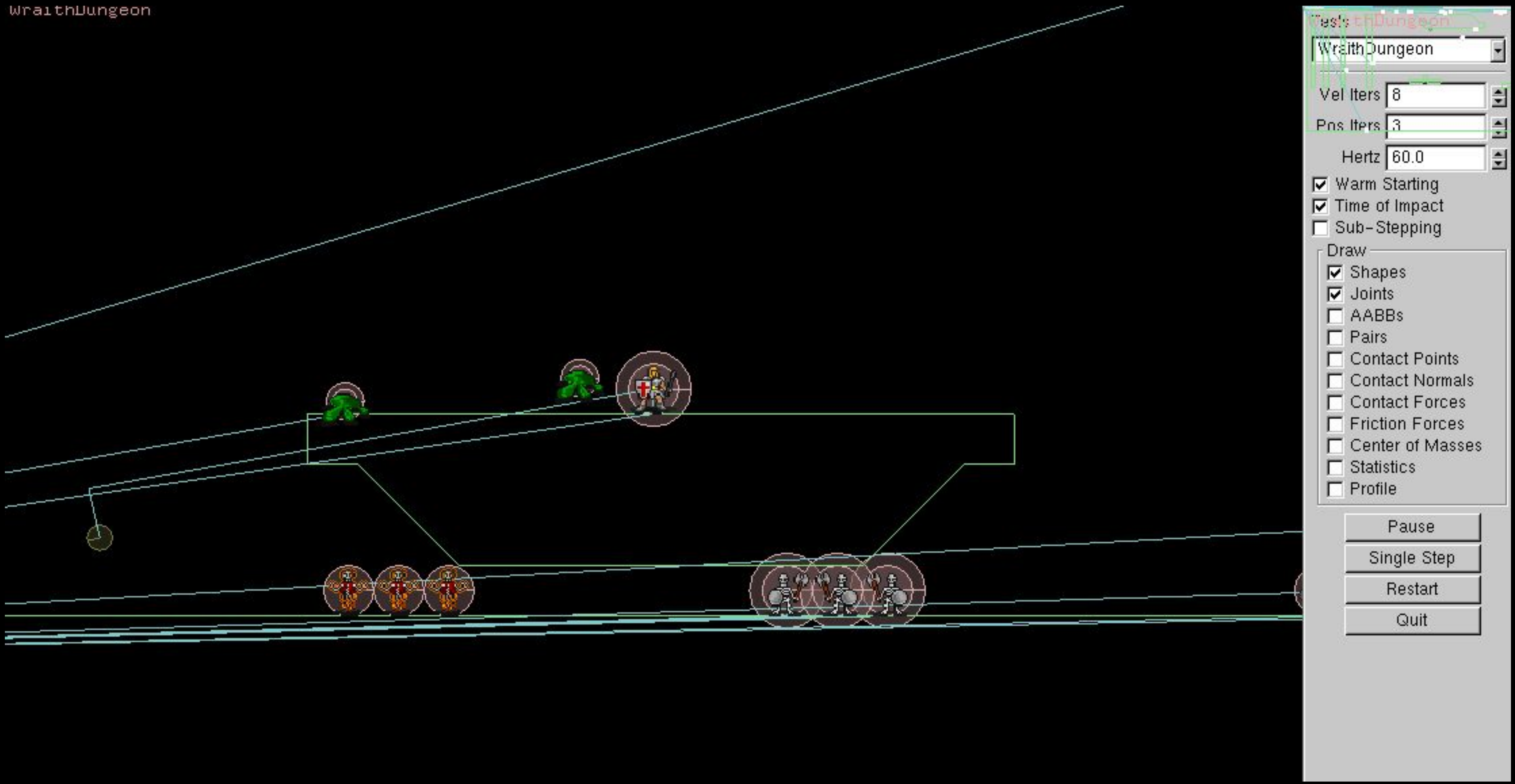

```
Current FPS: 13.9721
Average FPS: 13.4052
Worst FPS: 0.658328 1519 ms
Best FPS: 43.6941 16 ms
Triangle Count: 164924
Batch Count: 6109
```


```
FPS: 62    ARROW KEYS MOVE, SPACE JUMPS AND 'P' RESETS PLAYER POSITION
MX: 1608, MY: -482
```

First real attempt at 2d engine



Demo



WraithDungeon

Tests
WraithDungeon

Vel Iters 8
Pos Iters 3
Hertz 60.0

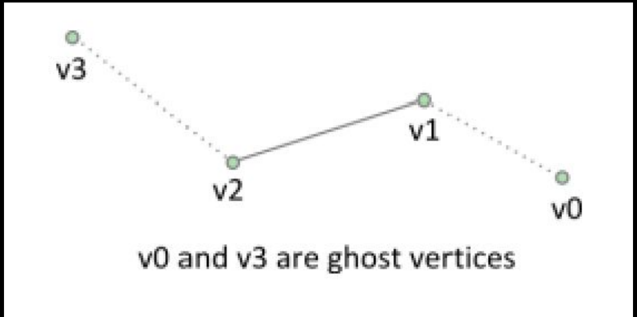
☑ Warm Starting
☑ Time of Impact
☐ Sub-Stepping

Draw
☑ Shapes
☑ Joints
☐ AABBs
☐ Pairs
☐ Contact Points
☐ Contact Normals
☐ Contact Forces
☐ Friction Forces
☐ Center of Masses
☐ Statistics
☐ Profile

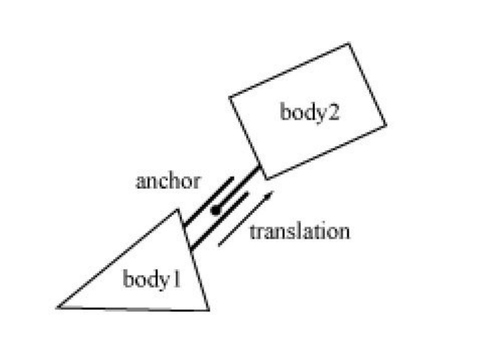
Problems

- Adjusting reactions indirectly (can fly by swinging mace)
- Ghost collisions with internal edges on tile map
- Projectiles get funky when map gets crowded
- Made hacky wall stick constraint that breaks on moving objects
- Difficult to detect solid areas

edge shapes to fix ghost collisions



hacked prismatic joint to make “wall stick” joint



Back to the Drawing Board!

(But this time armed with the concept of “speculative contacts”)

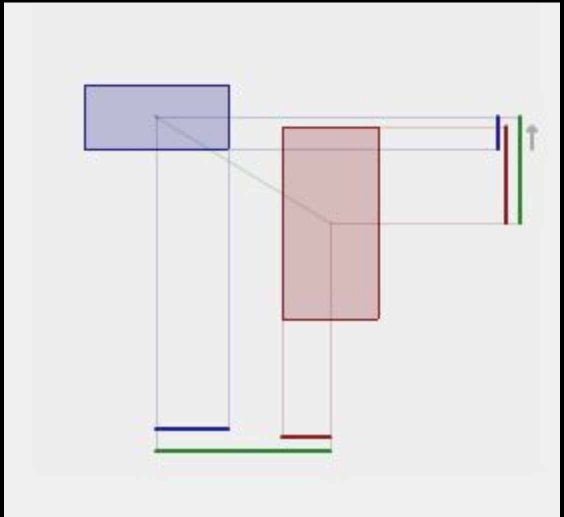
What are the building blocks of a collision engine?

- Broad Phase
  - Easy, many books and school text books will help you here
  - Does not affect dynamics of simulation in most cases
  - Can be switched out with better system for performance
  - So I use brute force until I get the narrow phase down
- Narrow Phase
  - Usually solved with many pairwise comparisons
  - Very tricky to get right
  - Not much information on how to do this until fairly recently
  - Most energy put into finding decent global solution between all bodies

Important Concepts

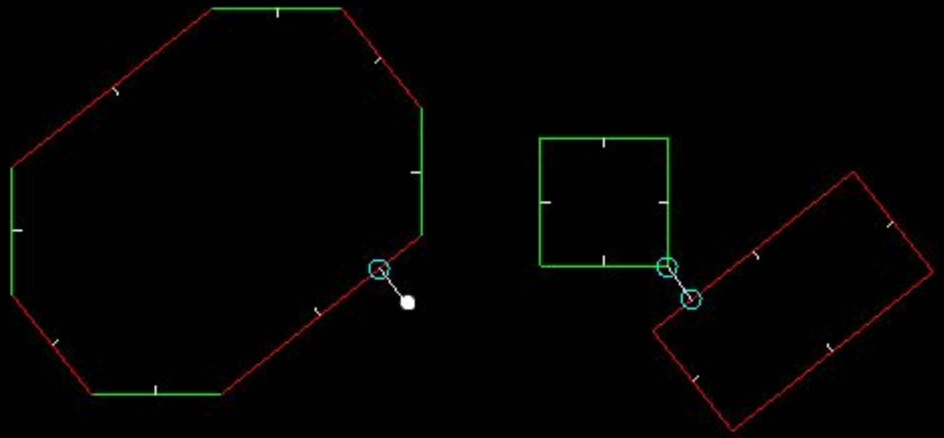
- Ray casting
- Separating axis theorem (SAT)/Minkowski difference
- Gilbert–Johnson–Keerthi distance algorithm
- Speculative contacts
- Ghost Collisions
- Polygon skins or outer hulls

Separating Axis Theorem (SAT)

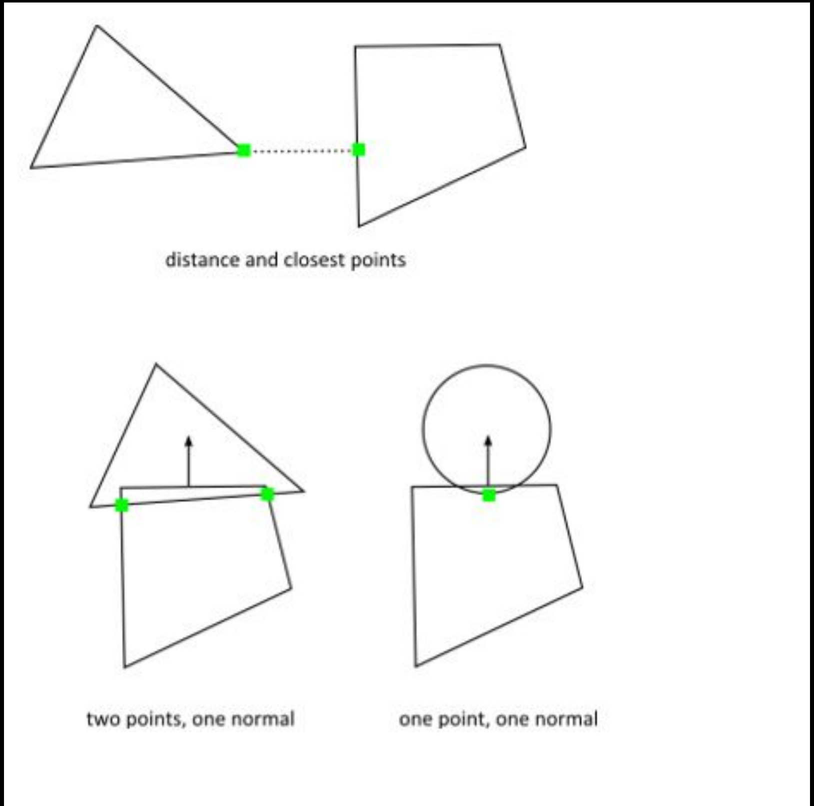


http://www.metanetsoftware.com/2016/n-tutorial-a-collision-detection-and-response

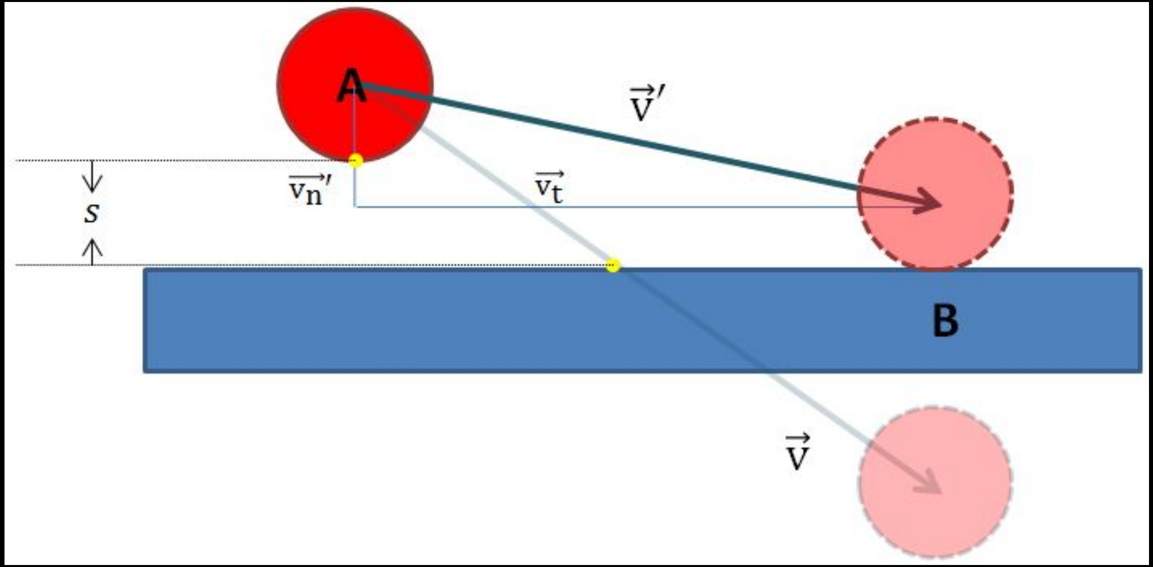
Minkowski Sum & Difference

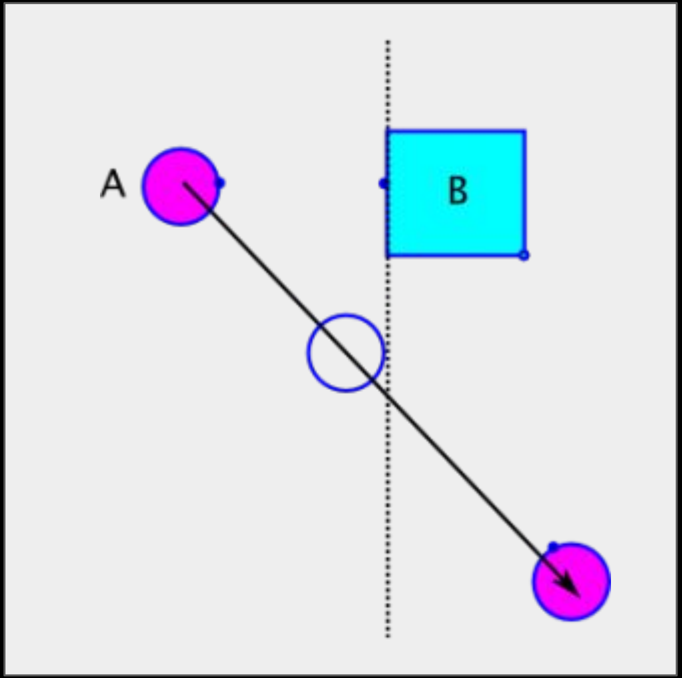


Gilbert–Johnson–Keerthi Distance Algorithm

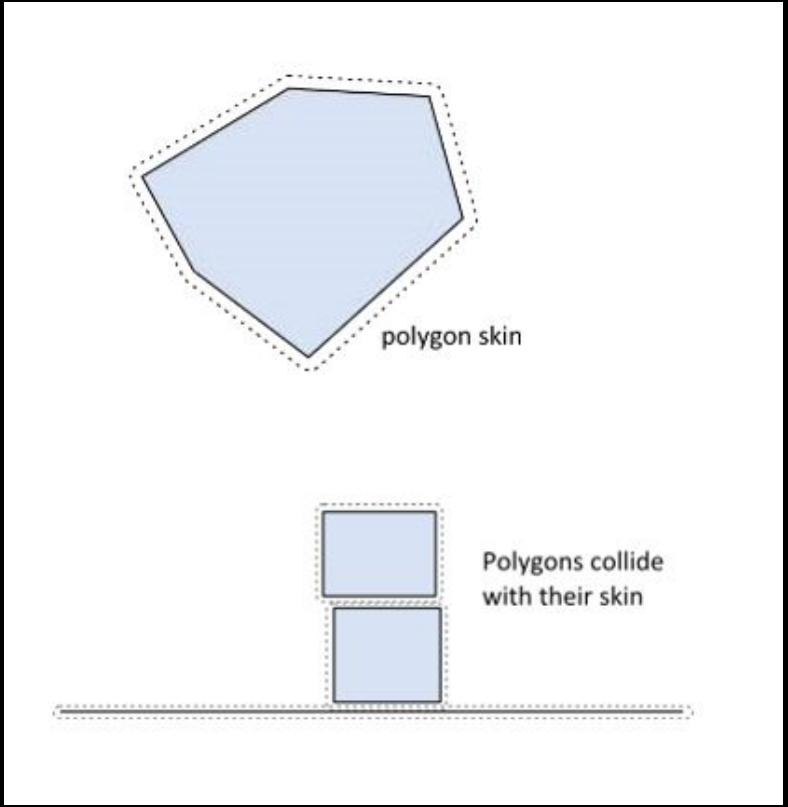


Speculative Contacts and Ghost Collisions

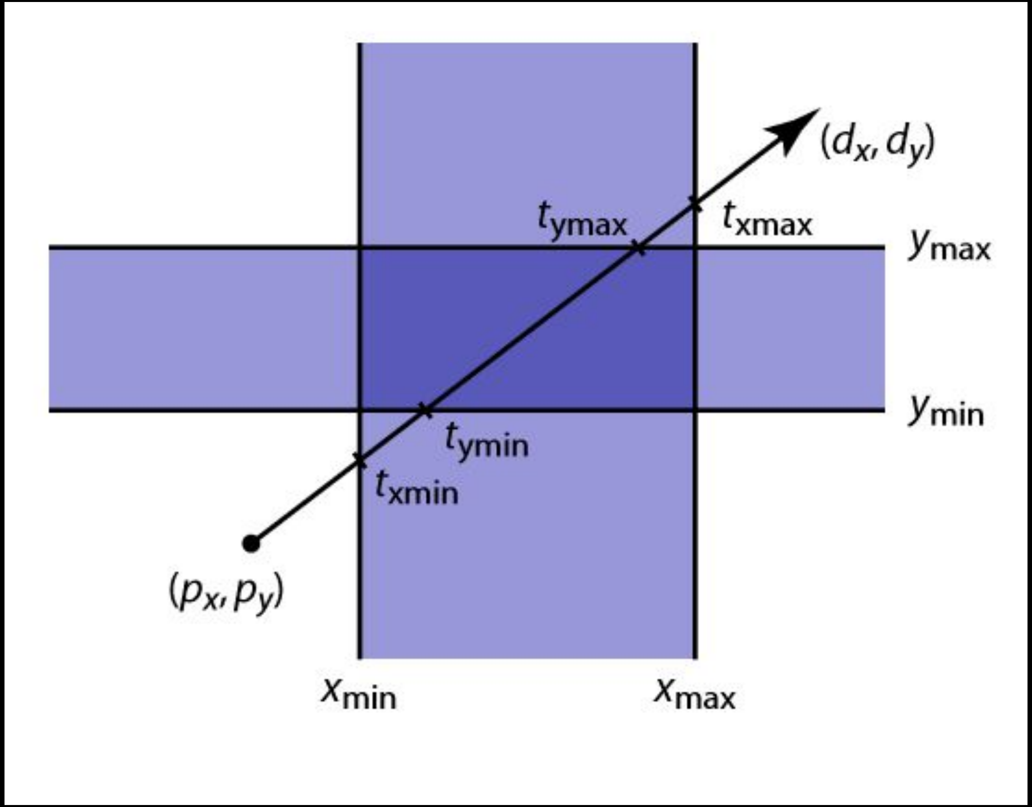




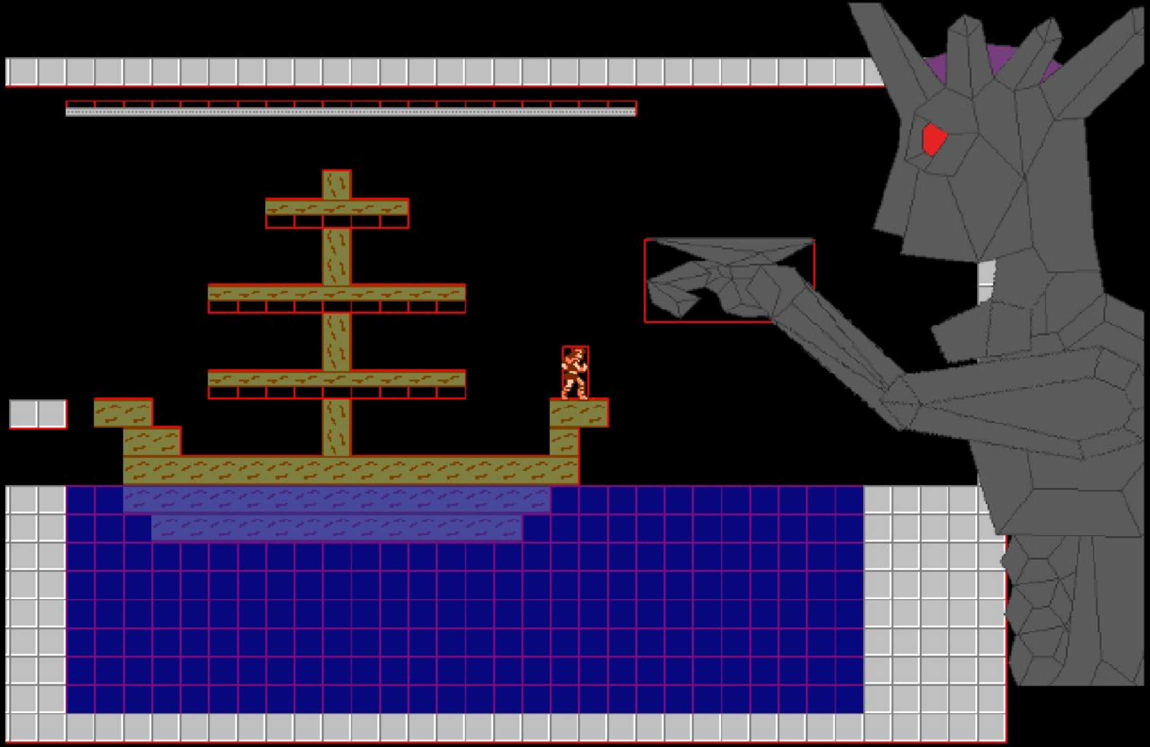
Polygon Skins or Outer Hulls



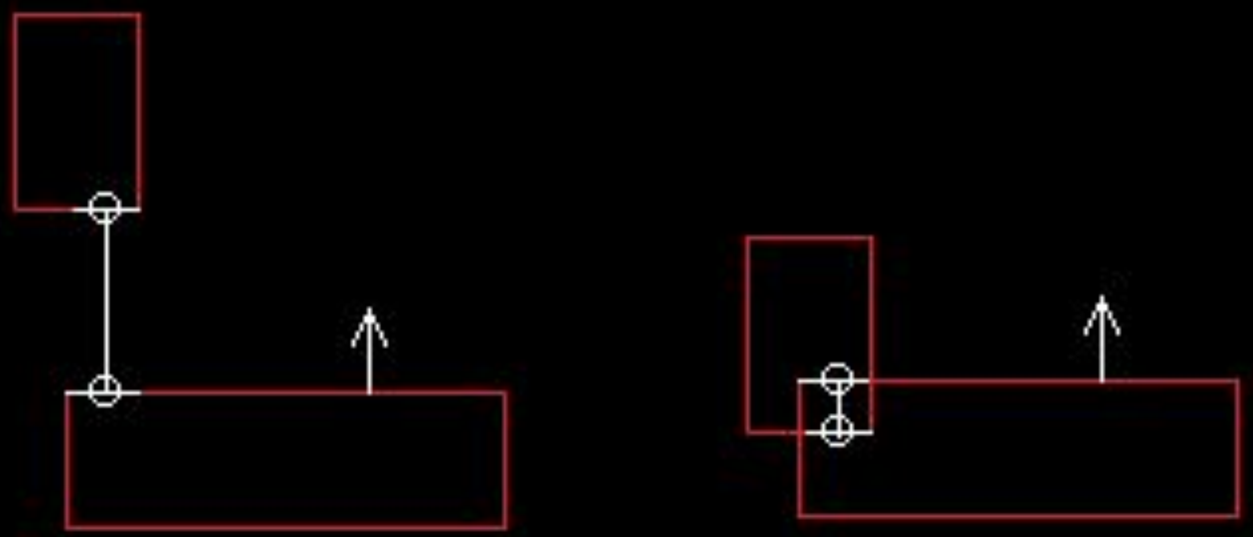
Ray Casting



Anatomy of a Collision Engine

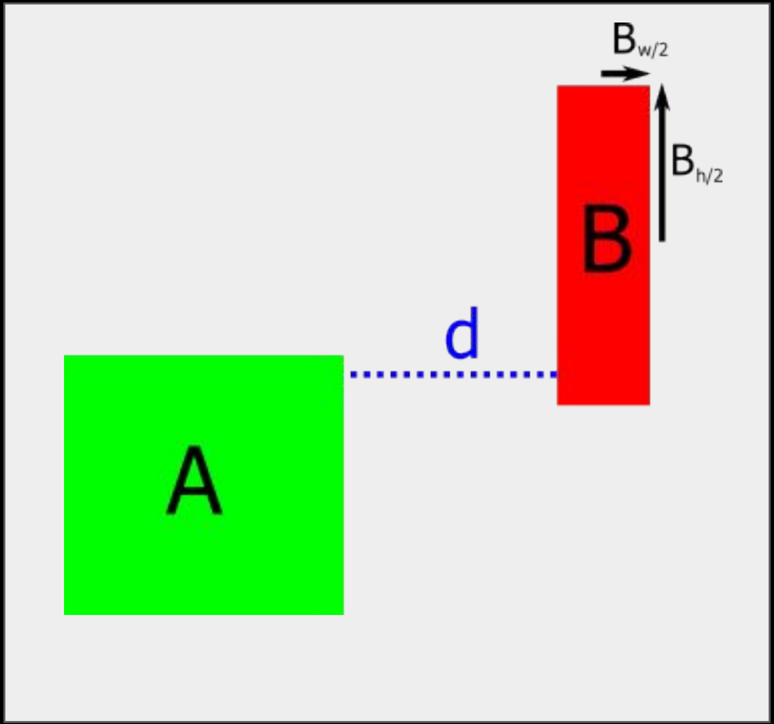


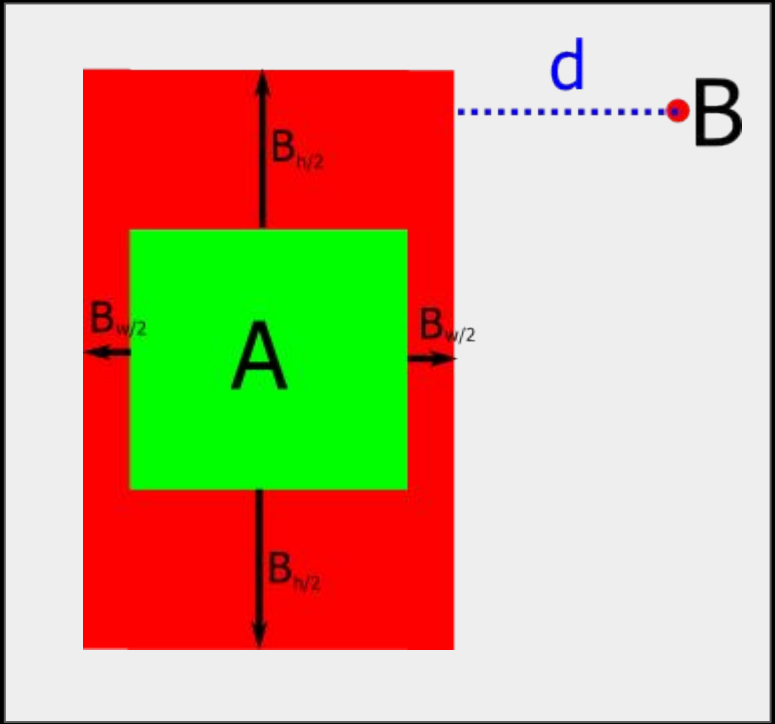
“Distance Query”



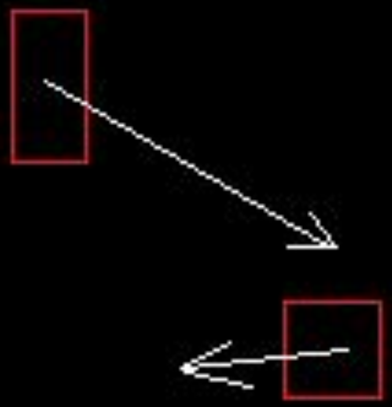
Normal, Distance (separation is negative), Manifold

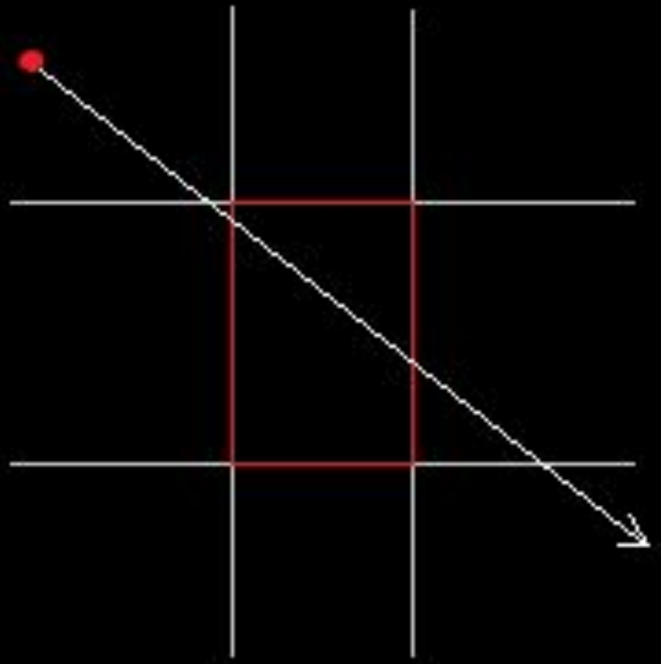
“Sweep”





“Sweep”

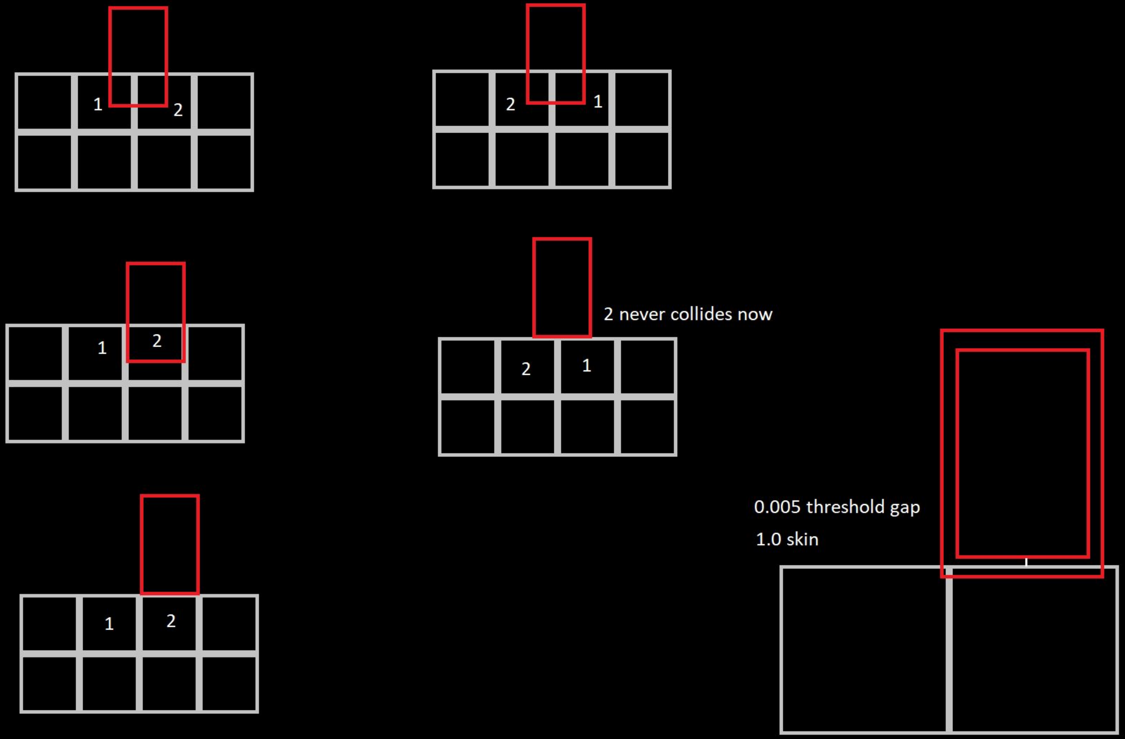




Global Solution

- Object priorities used for determining influence over a position
- Contacts sorted by priority and Y-Axis for stacking
- Box “skins” used to prevent falling through platforms
- Positioning and Contact resolution separated to stop ghost collisions

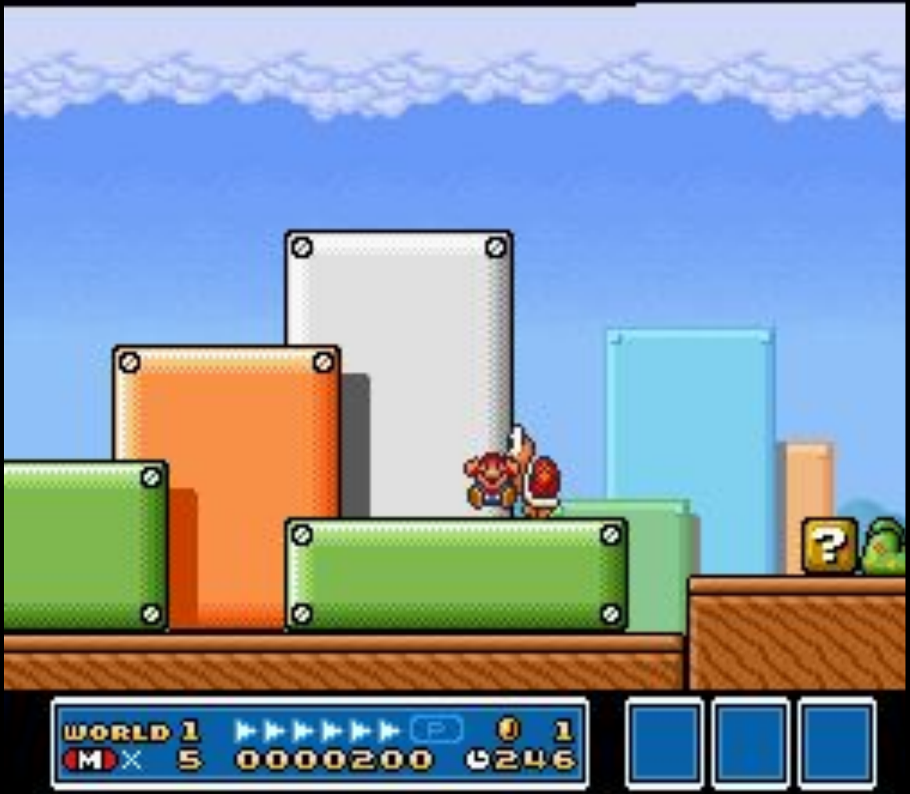
Internal Edges

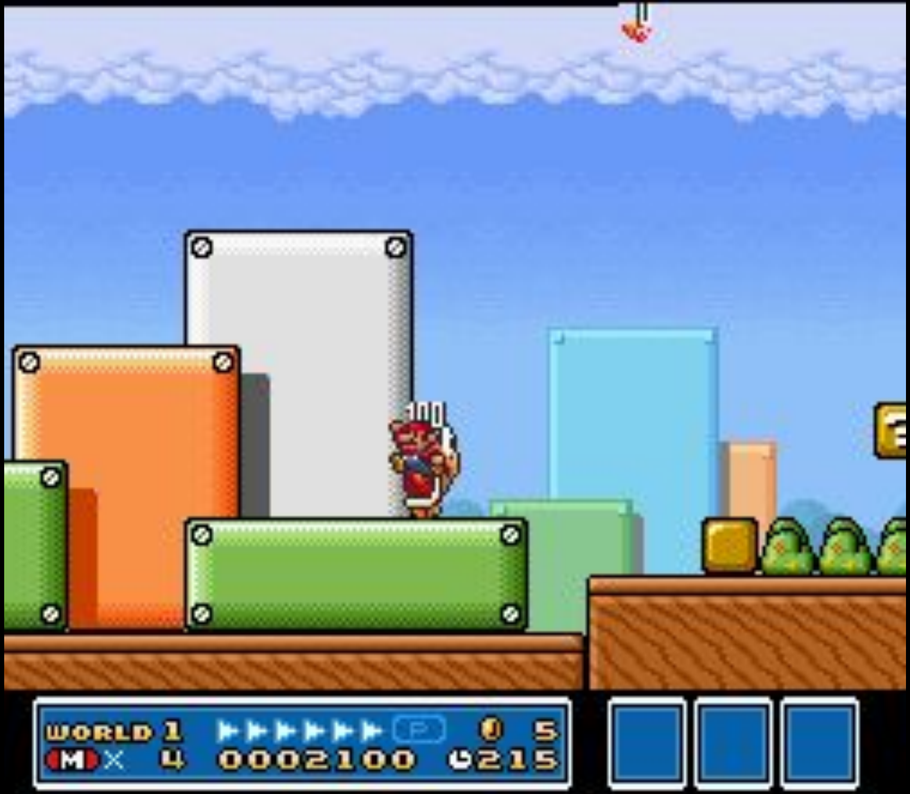


Main Loop

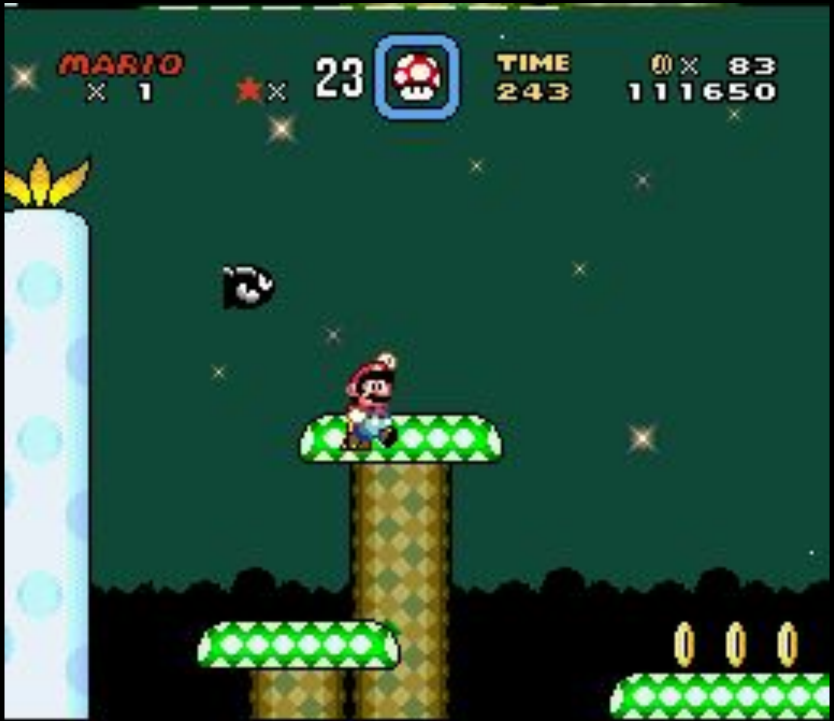
- Set body displacements to current velocities
- Clip displacements with previous frame contacts to avoid internal edge hits
- Generate contact pairs
- Adjust displacements using speculative contacts and ray casts
- Integrate and move bodies position using displacement
- Generate current frame contacts using distance queries
- Allow user to adjust velocities any way they like before going back to top of loop, including using physics joint resolutions

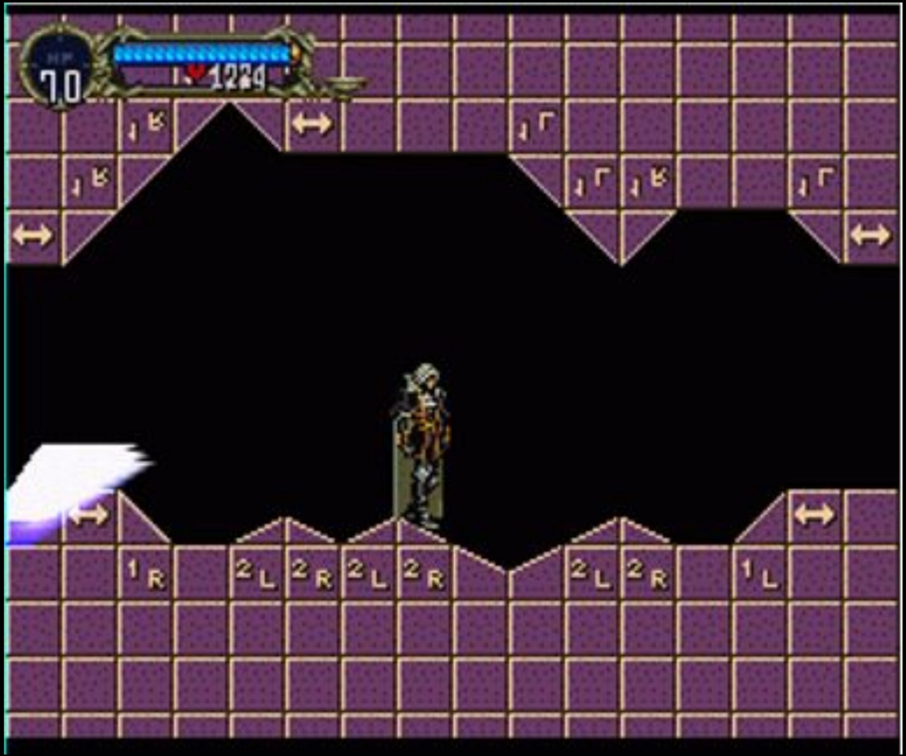
The Difference Between Life and Death!



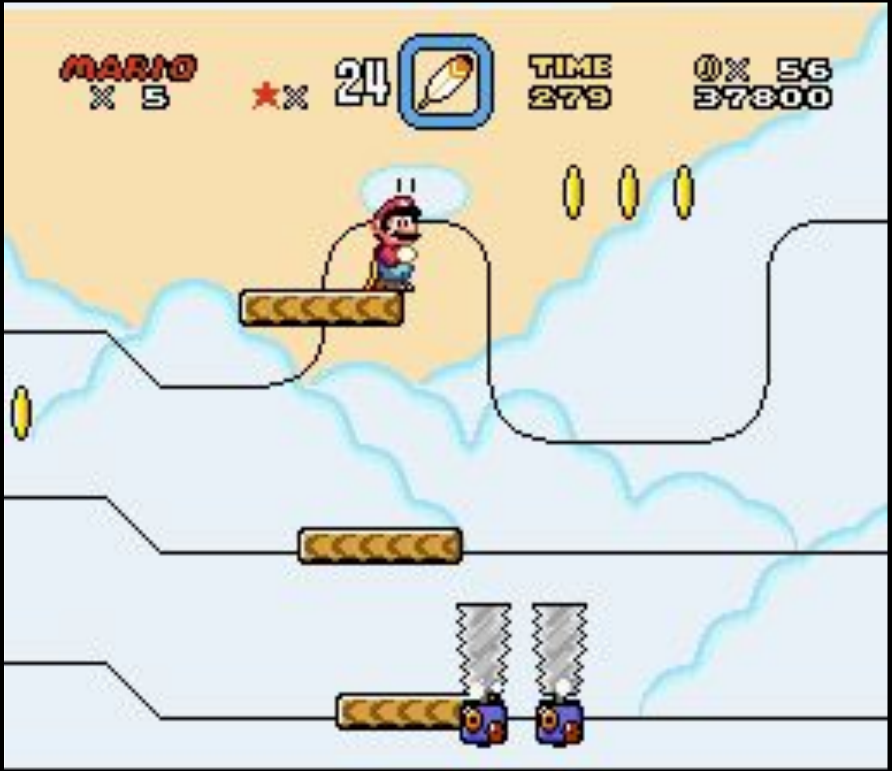


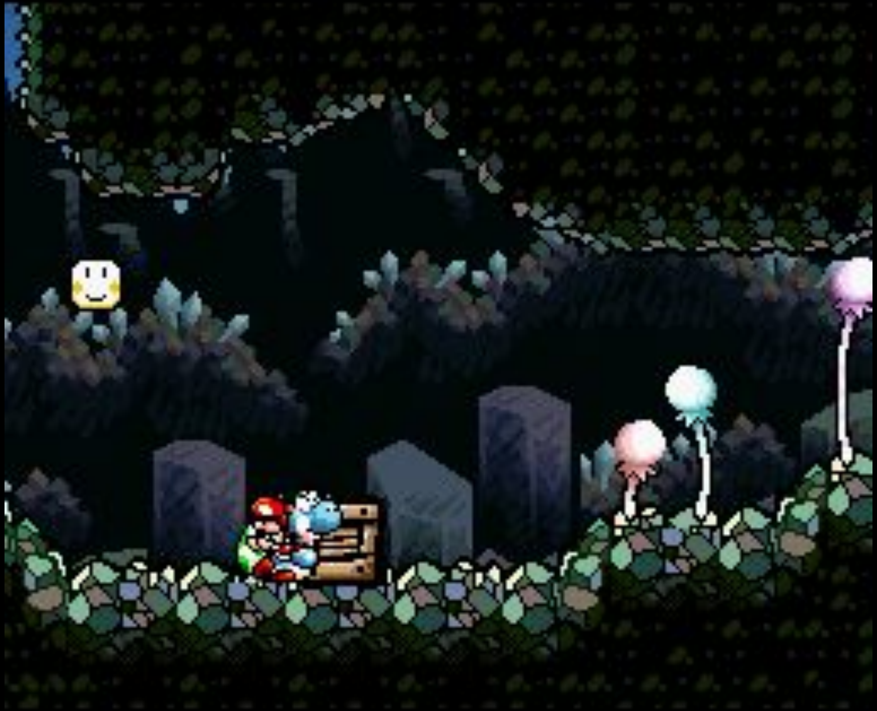
One Way Platforms, Inclines



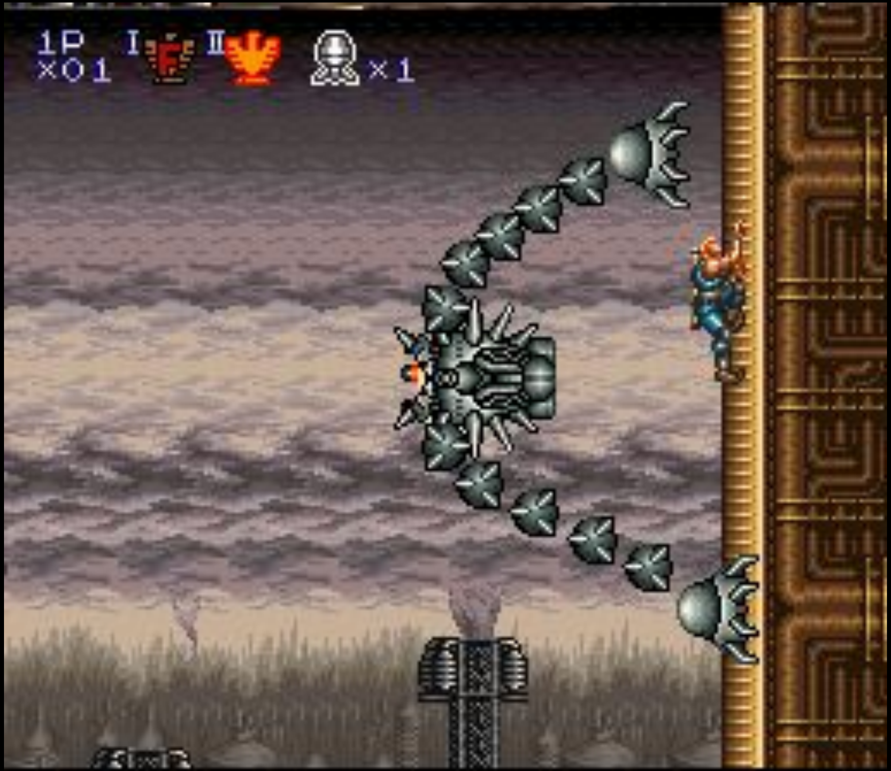


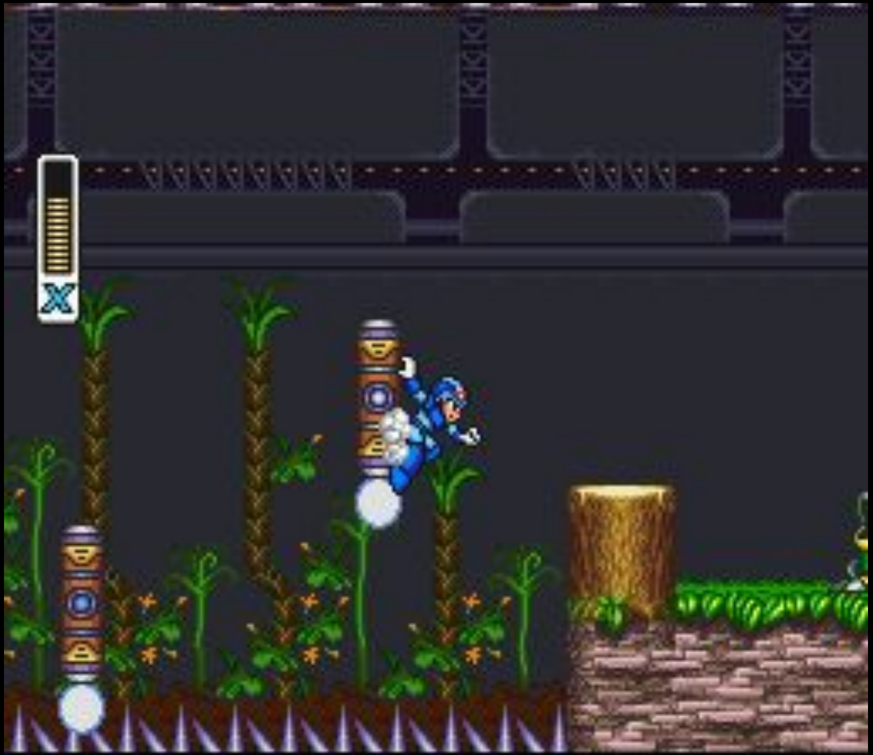
Moving Platforms, Pushable Blocks



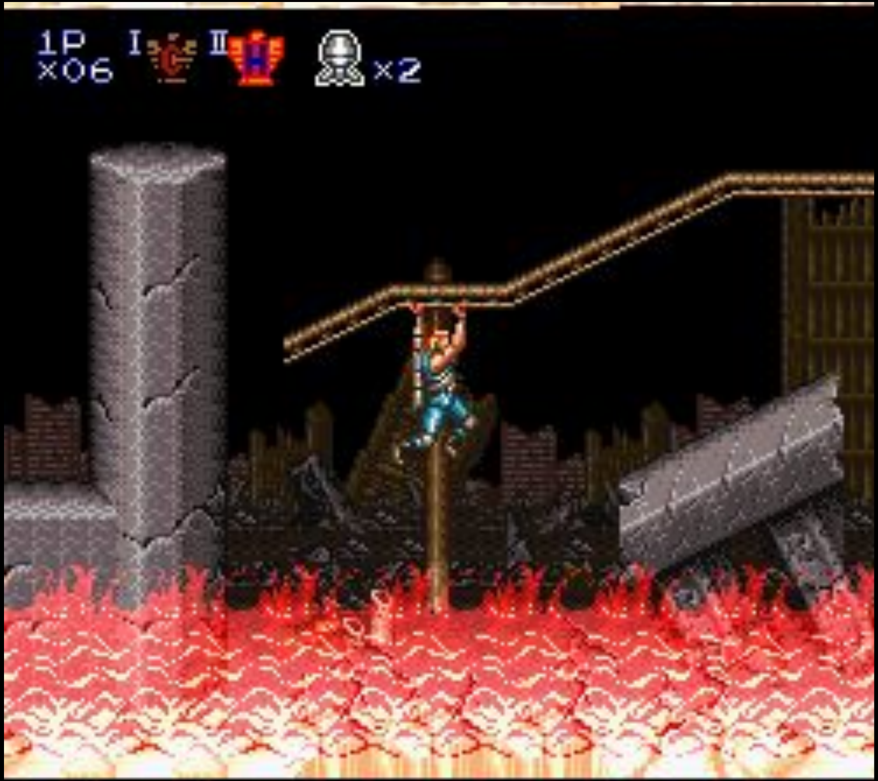


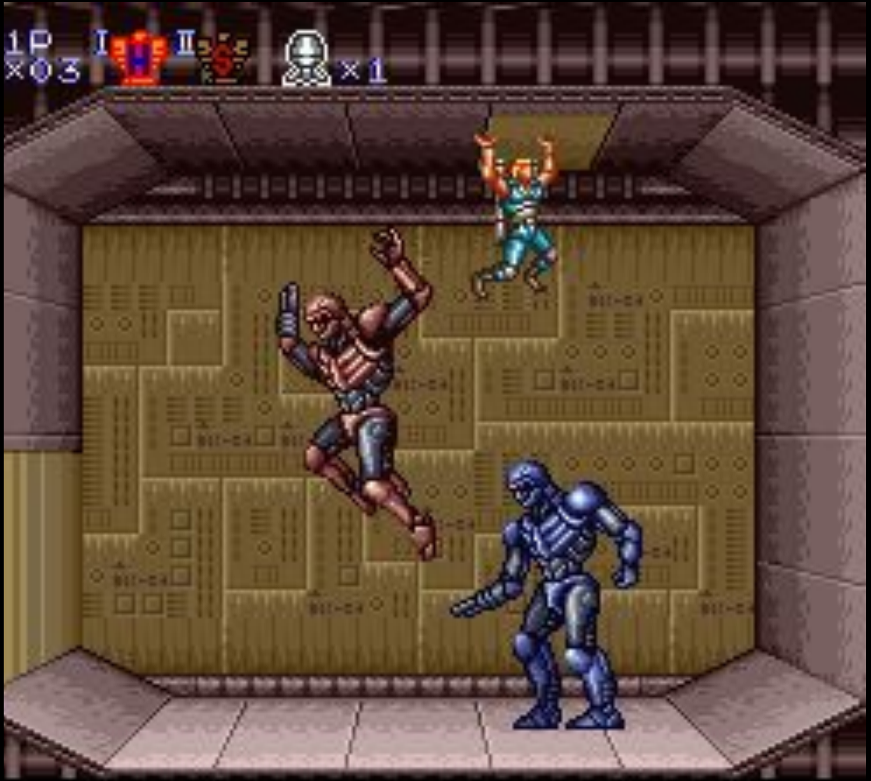
Wall Climbing, Wall Sliding



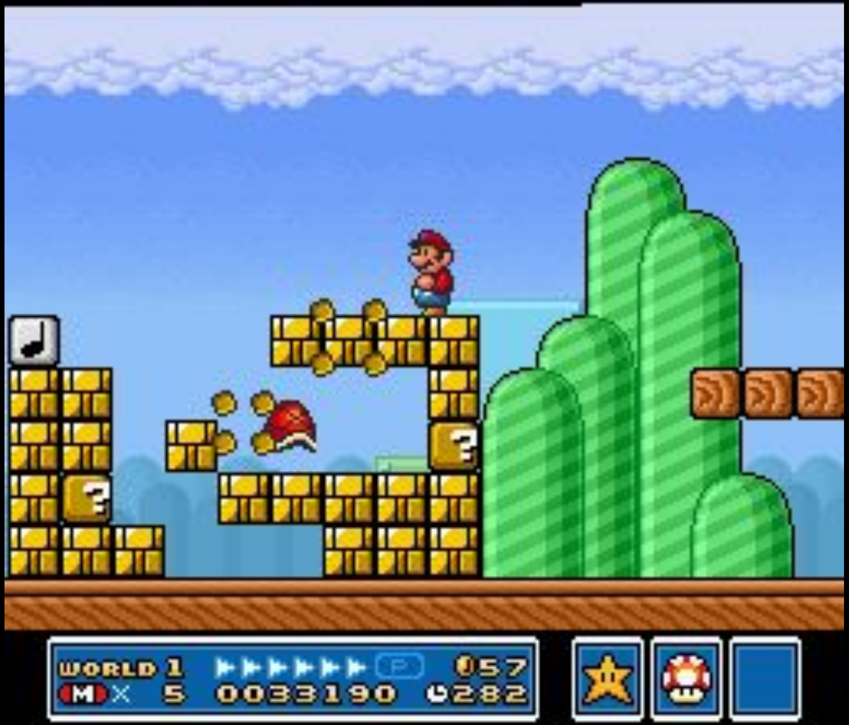


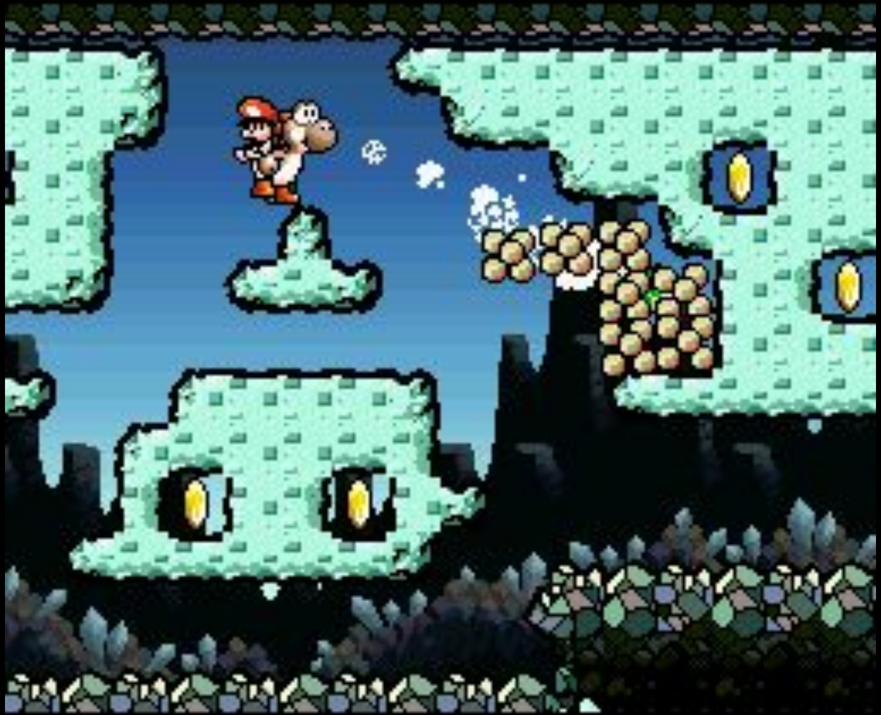
Hanging from bars or ceilings



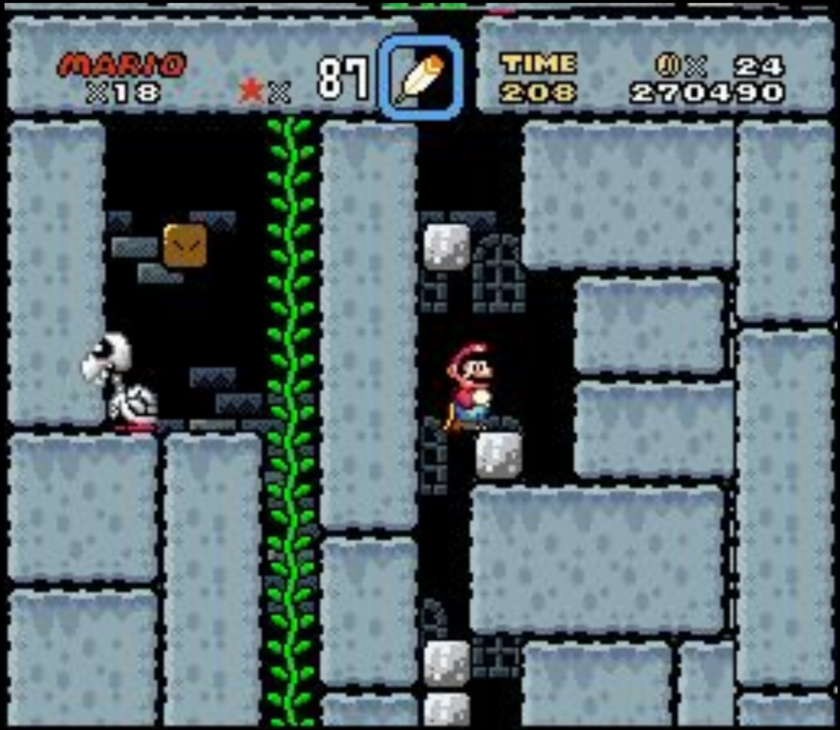


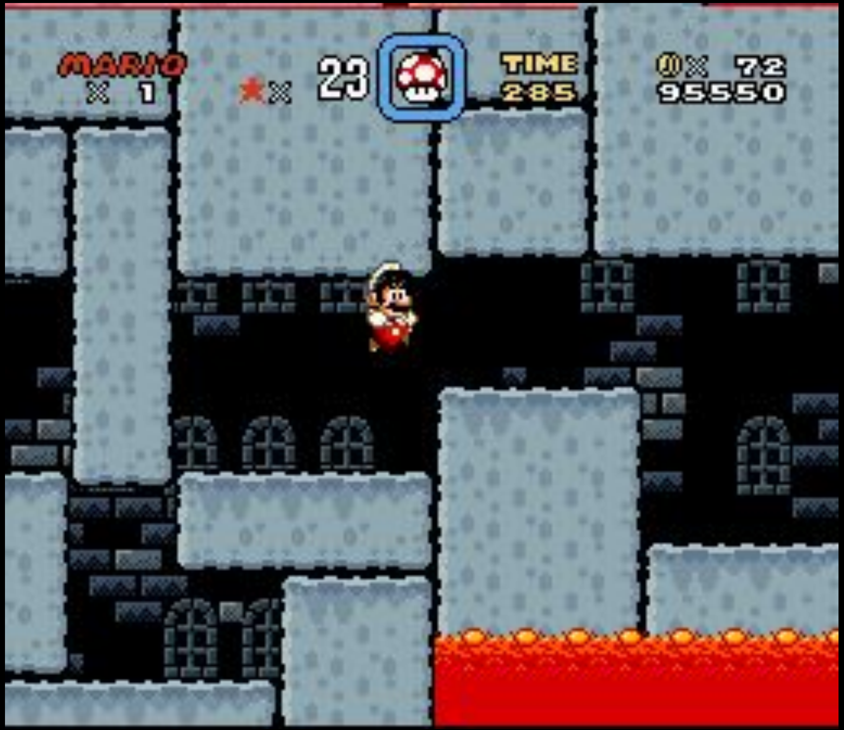
Dynamic Destructible Blocks



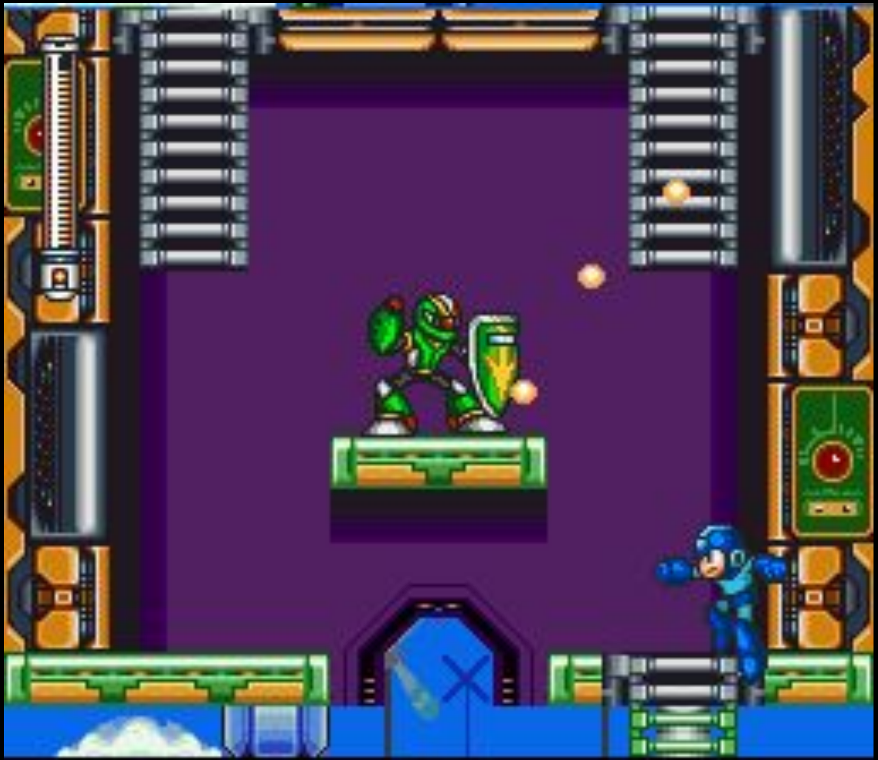


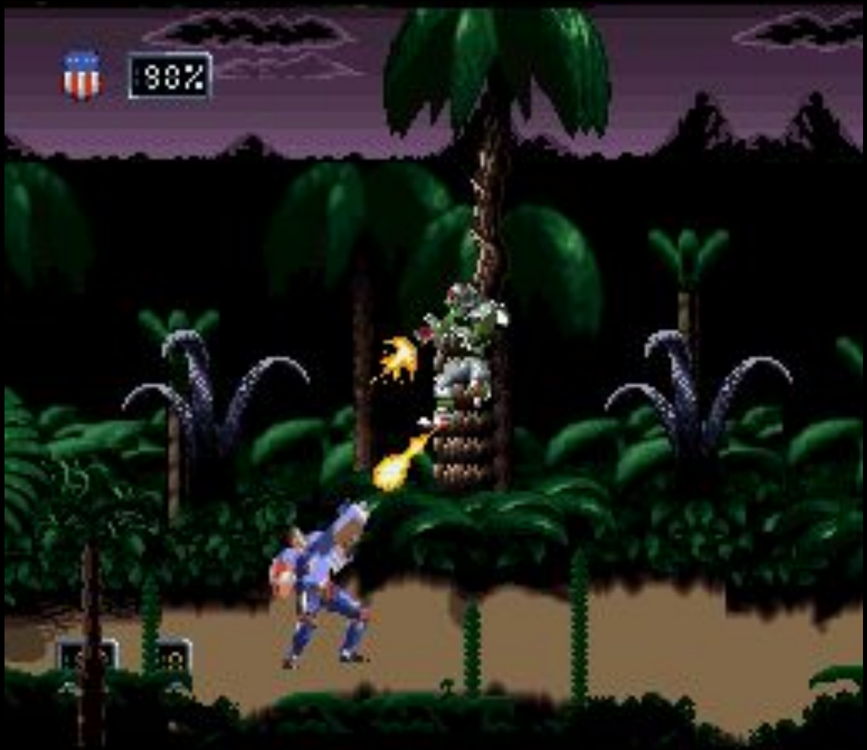
Moving Tilemaps



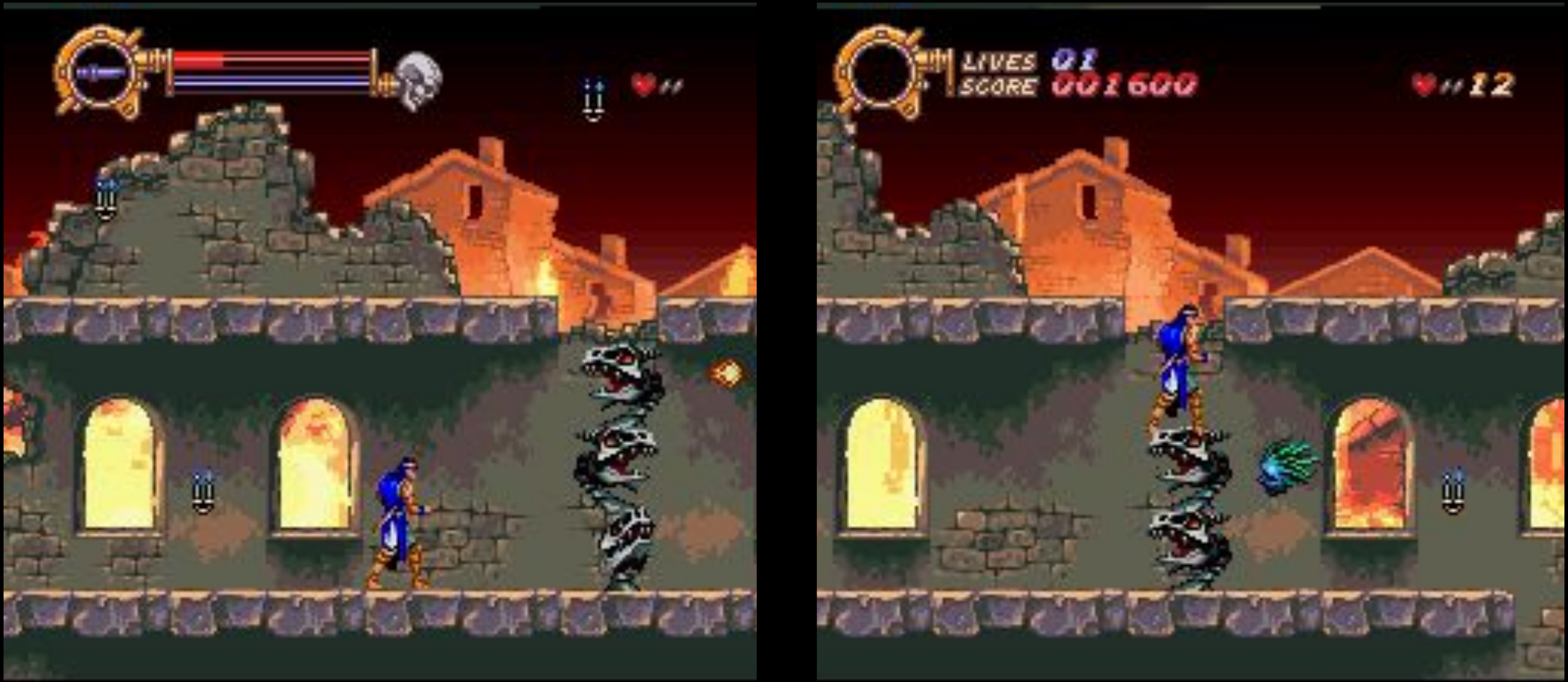


Shields, Fast Projectiles

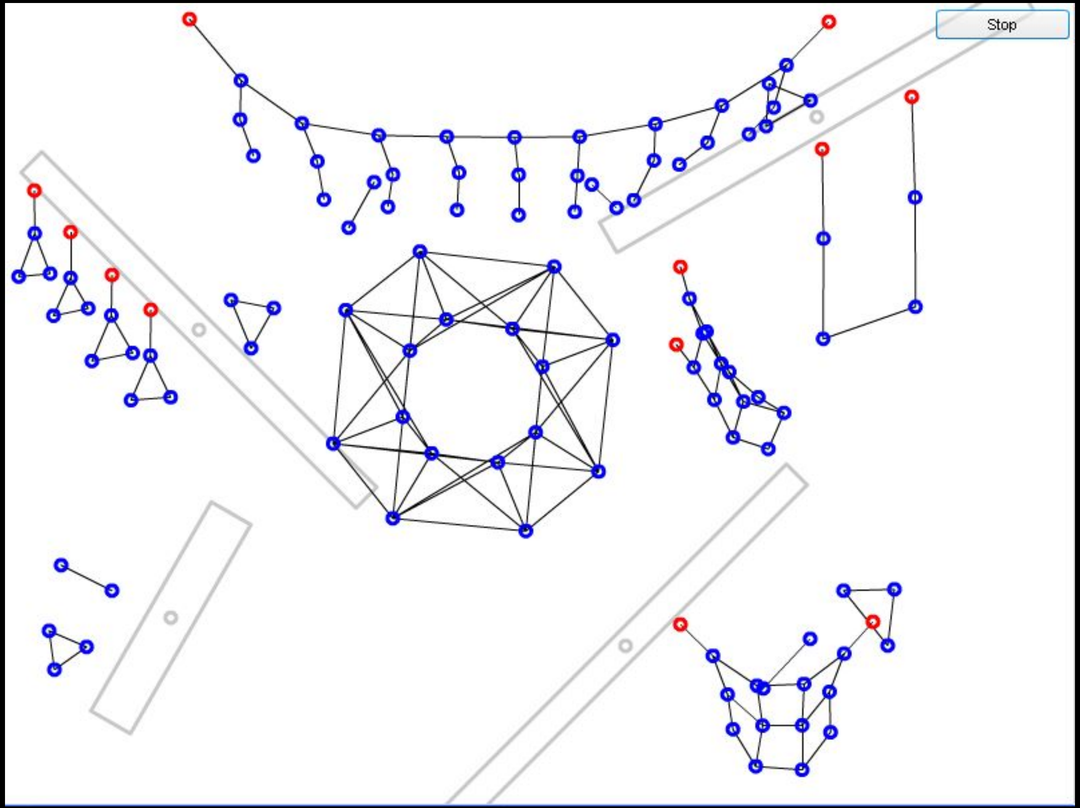




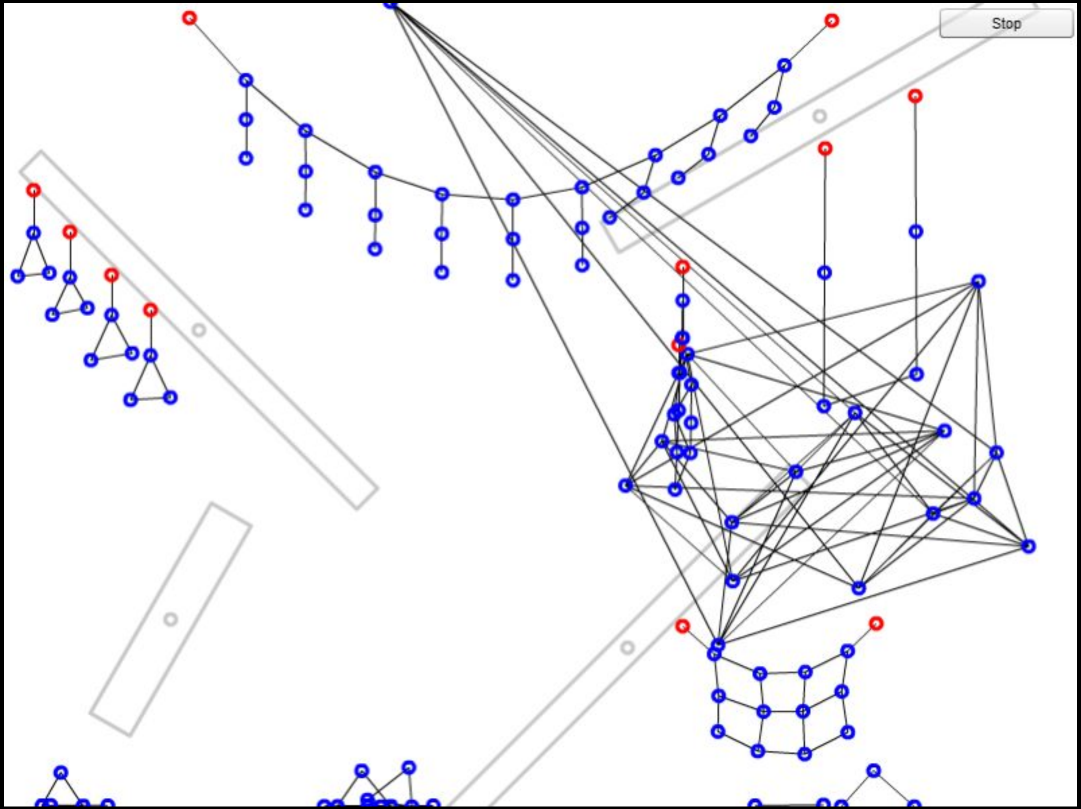
Dynamic Stacked Objects



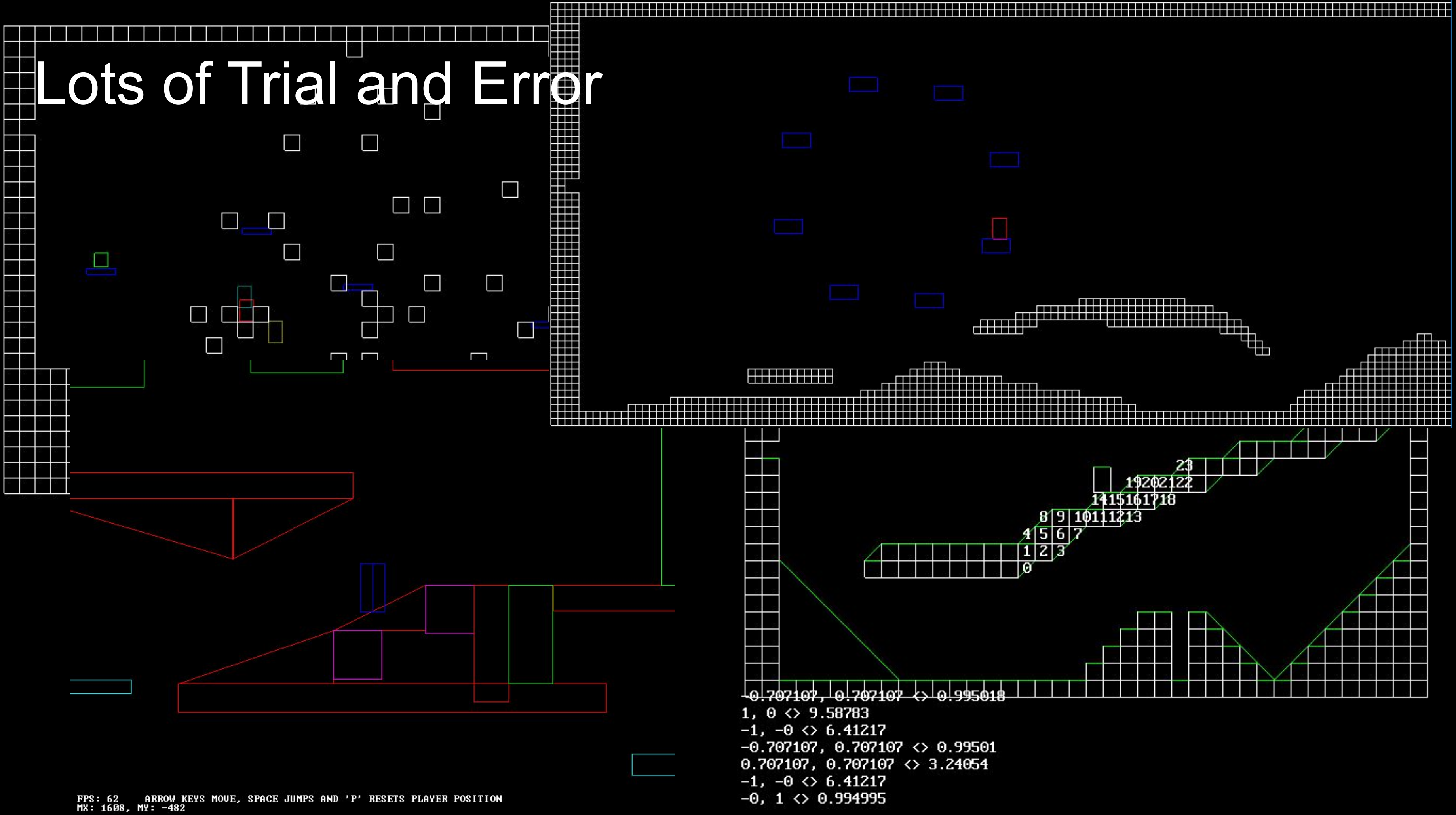
Joint Constraints



High School Physics with Euler Integration



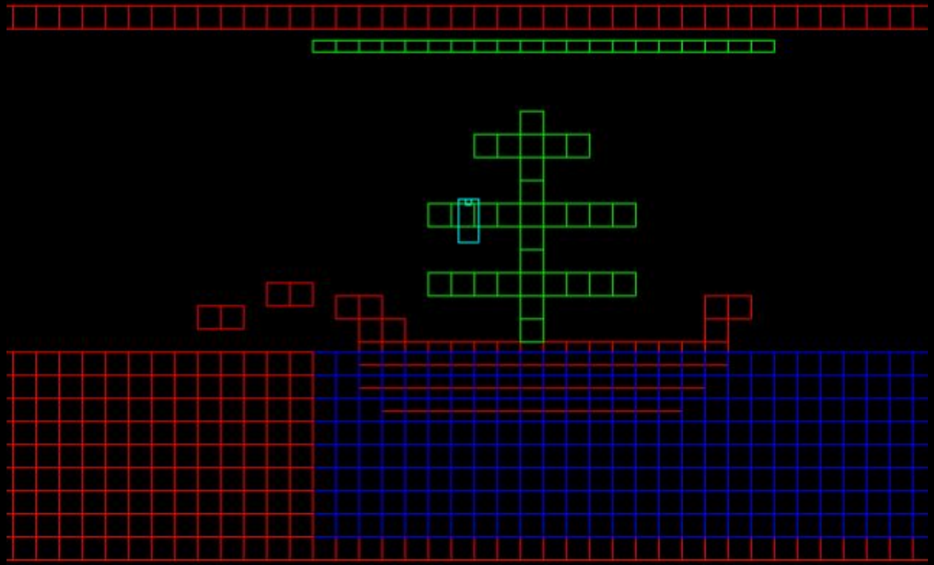
Lots of Trial and Error

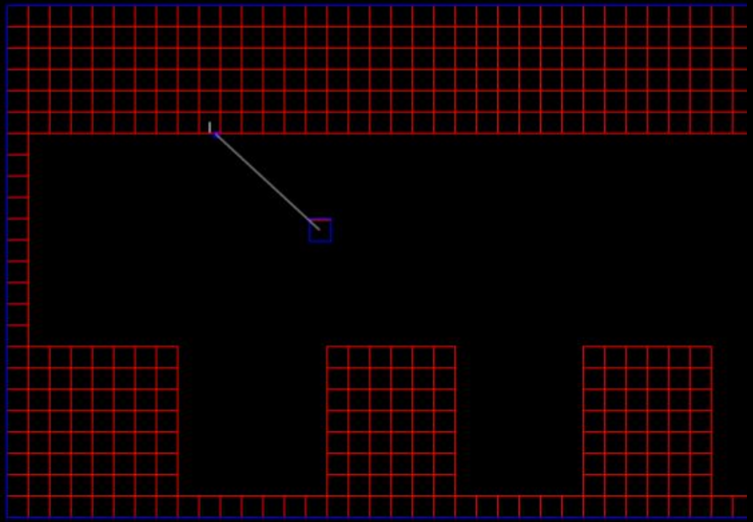


```
-0.707107, 0.707107 <> 0.995018
1, 0 <> 9.58783
-1, -0 <> 6.41217
-0.707107, 0.707107 <> 0.99501
0.707107, 0.707107 <> 3.24054
-1, -0 <> 6.41217
-0, 1 <> 0.994995
```

```
FPS: 62    ARROW KEYS MOVE, SPACE JUMPS AND 'P' RESETS PLAYER POSITION
MX: 1608, MY: -482
```

On With The Demo!

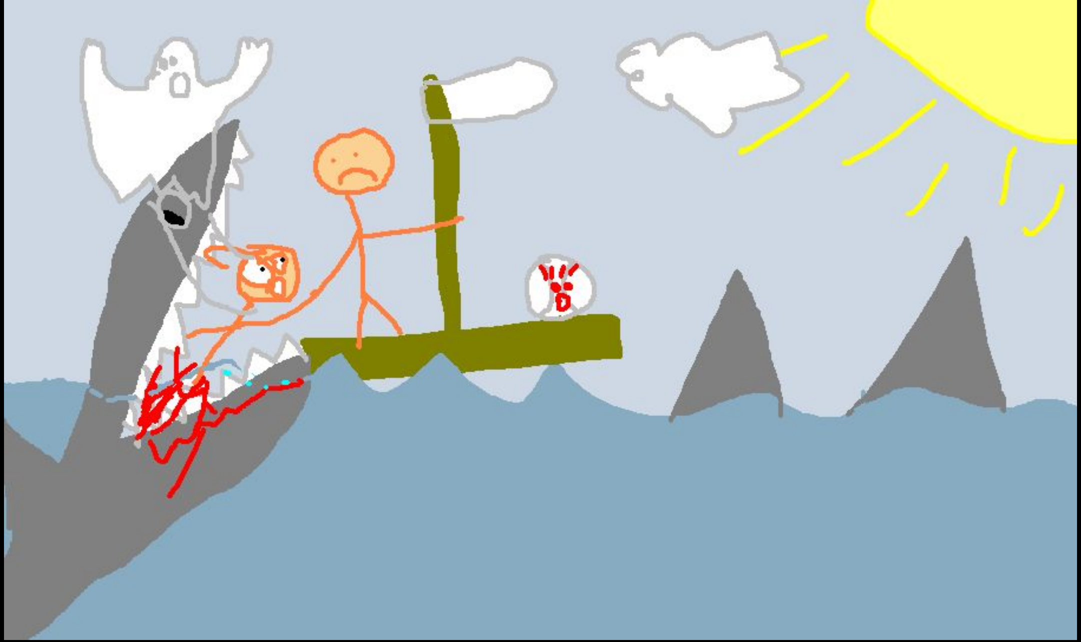




Future Additions

- Inclines (not implemented yet)
- Oriented Bounding Boxes
- More Joints
- Use better language like C++ or C#
- Make version with more shape options
- Figure out what can be done in parallel

Links and References



http://www.metanetsoftware.com/

http://box2d.org/

https://wildbunny.co.uk/

http://bulletphysics.org/wordpress/

https://github.com/mattleibow/jitterphysics

jeaton.matero.net/ ← my personal site!

http://www.cs.cornell.edu/courses/cs4620/2013fa/lectures/03raytracing1.pdf