

# 2D Shape Collision Simulation



By: Ken Dopp, Tikhon Jelvis,  
Gregory Nisbet, Jacob Taylor

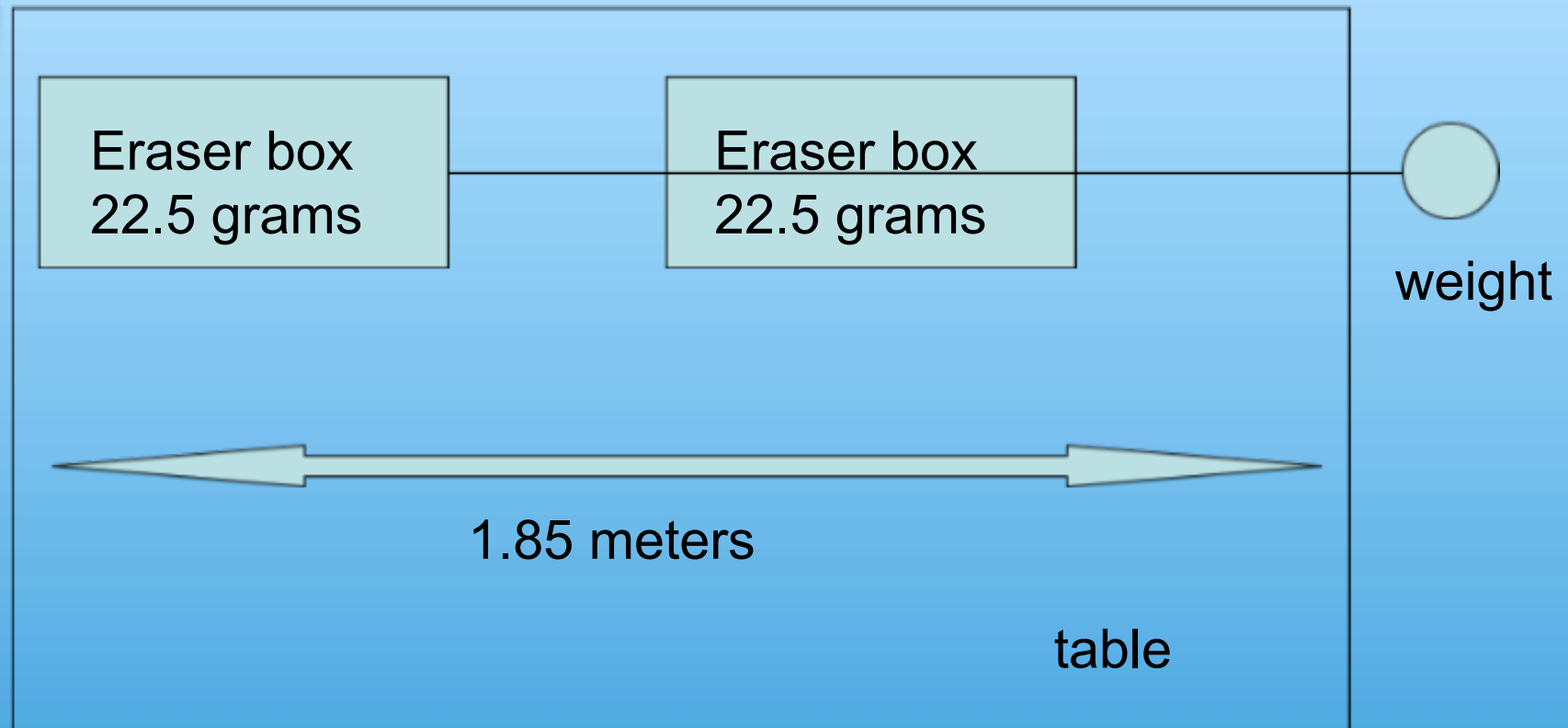


# Approach

## Two aspects:

- Physical collision experiment with eraser boxes
  - provides real world data to corroborate simulation
- Interactive computer simulation
  - Simulates collisions with a variety of shapes

# Experiment Setup





# Experimental Variables

Independent:

- Initial distance and orientations

Dependent:

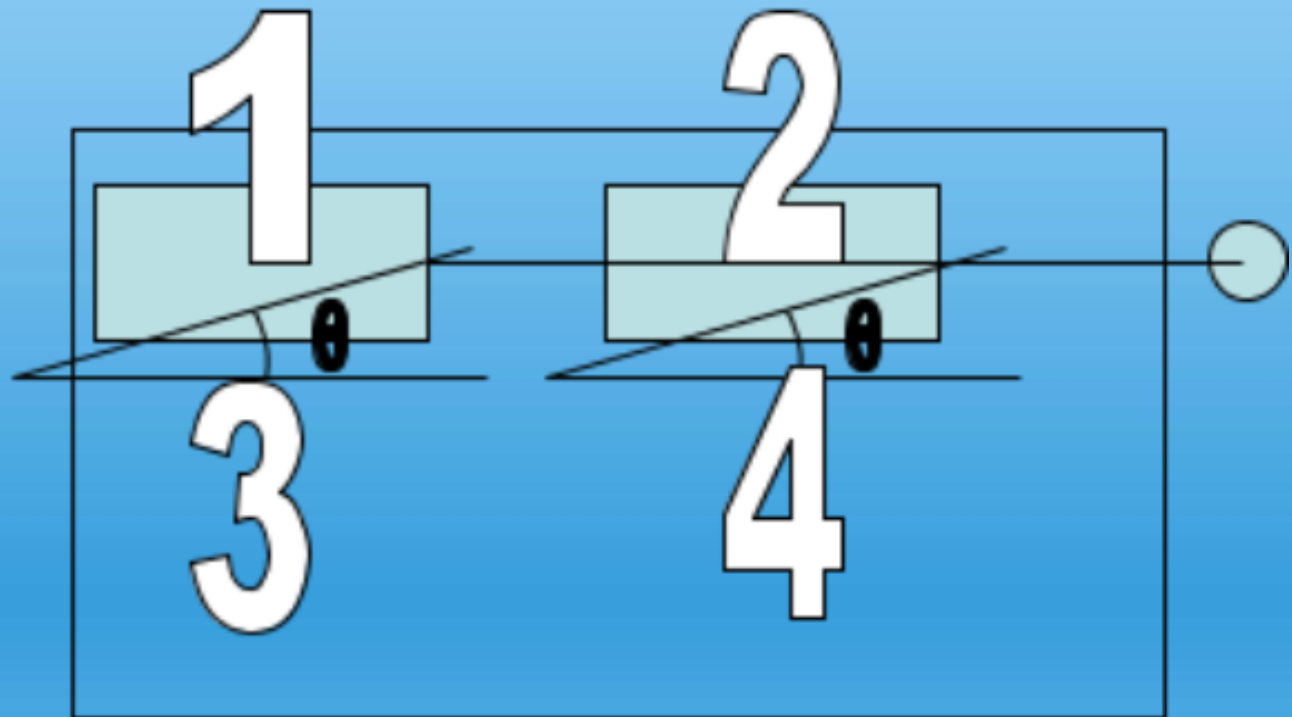
- Angles of objects
- Location of objects

Controlled:

- Force applied to object
- Friction of table.

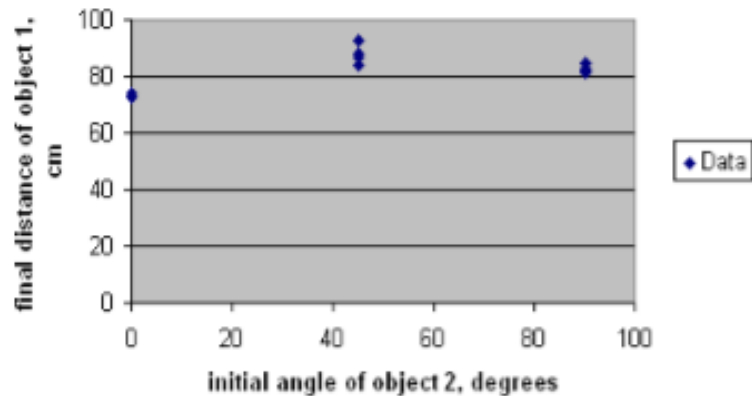
# Measurements

1. Distance of first box
2. Distance of second box
3. Angle of first box
4. Angle of second box

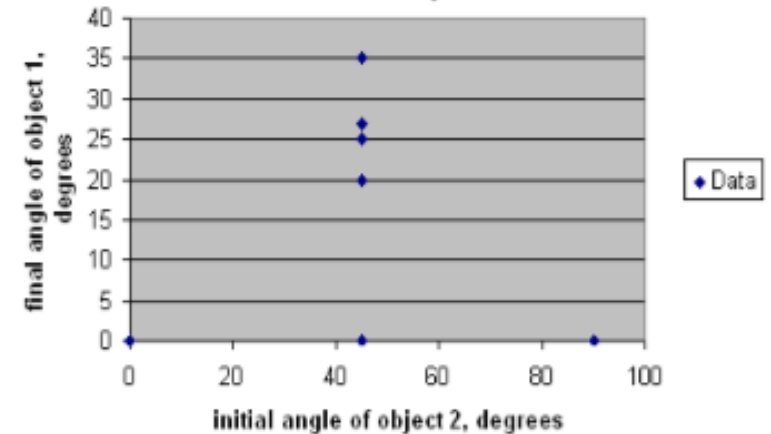


# Data

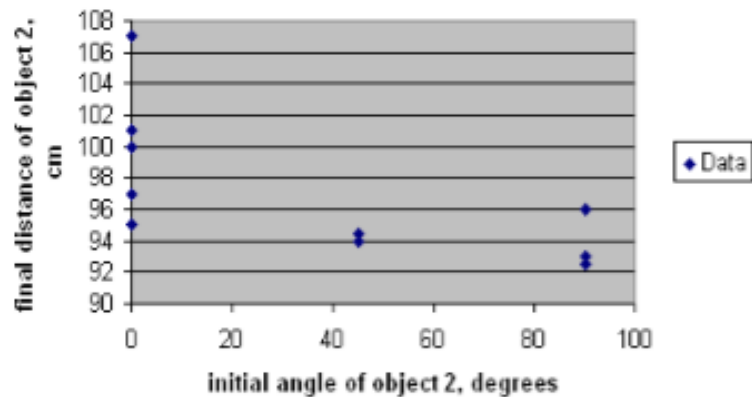
Effect of initial angle of object 2 on final distance of object 1



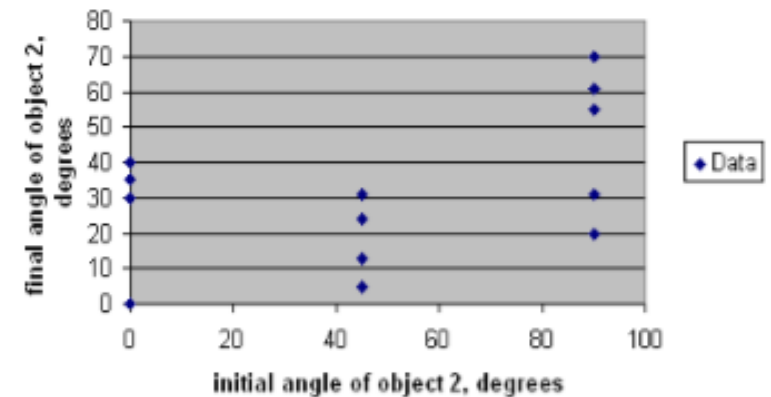
Effect of initial angle of object 2 on final angle of object 1



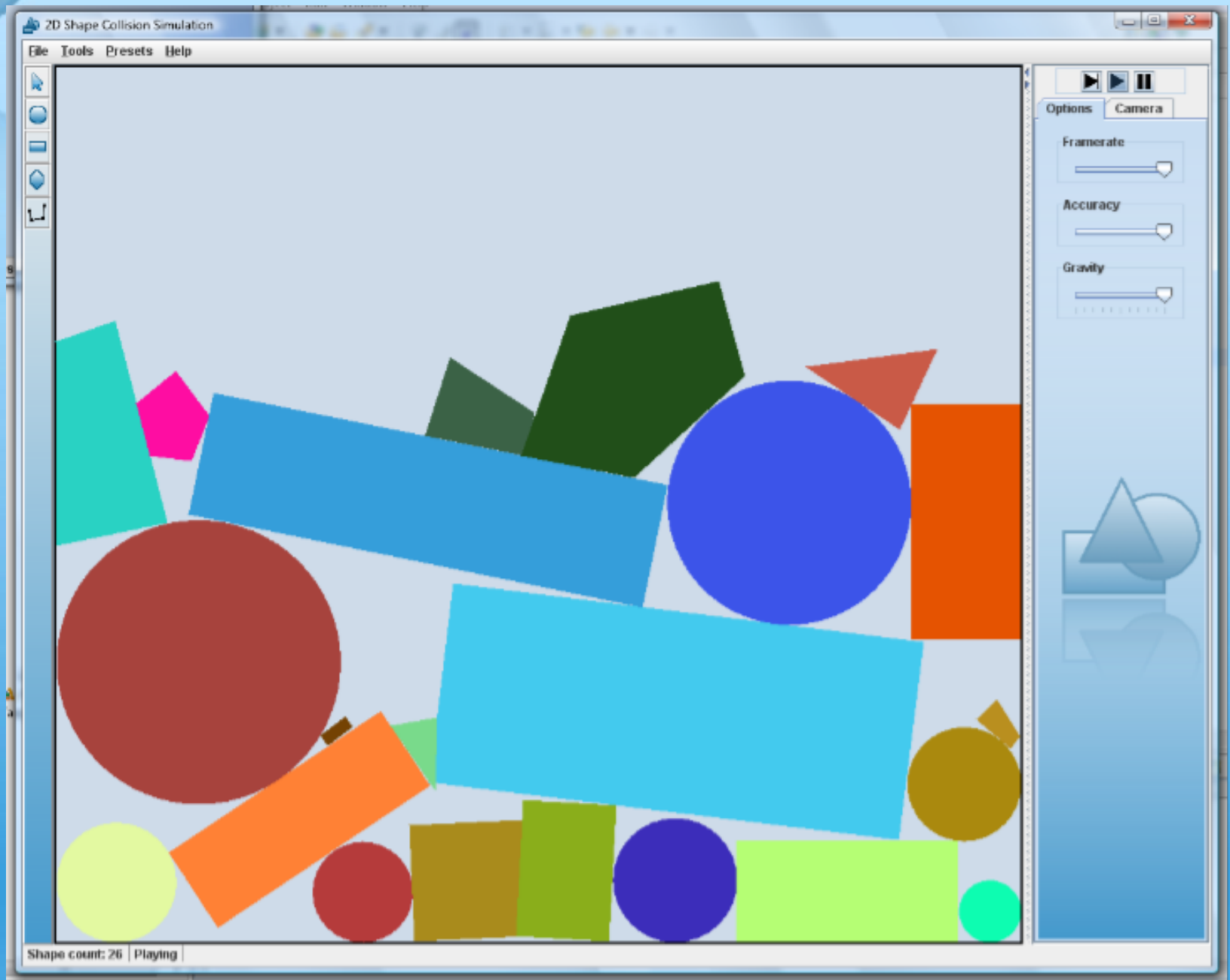
Effect of initial angle of object 2 on final distance of object 2



Effect of initial angle of object 2 on final angle of object 2



# Simulation





# Engine

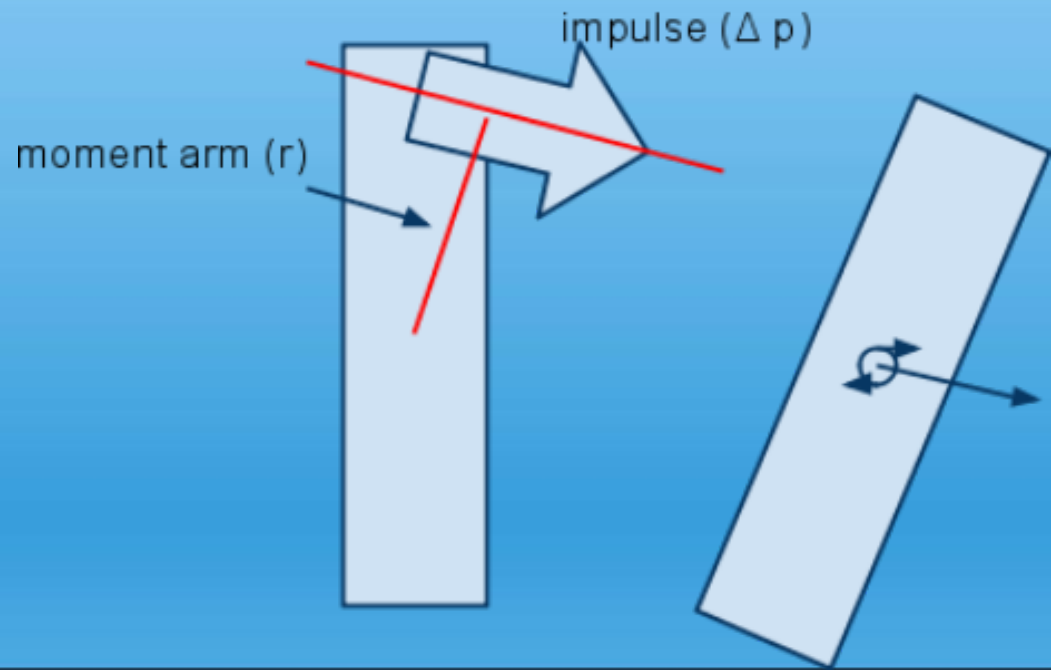
- Contained in one package (separate from the GUI)
- Contains circles and polygons
- Each shape has a position, velocity, angular velocity, mass, and moment of inertia
  - Moment of inertia : measure of how difficult an object is to turn.





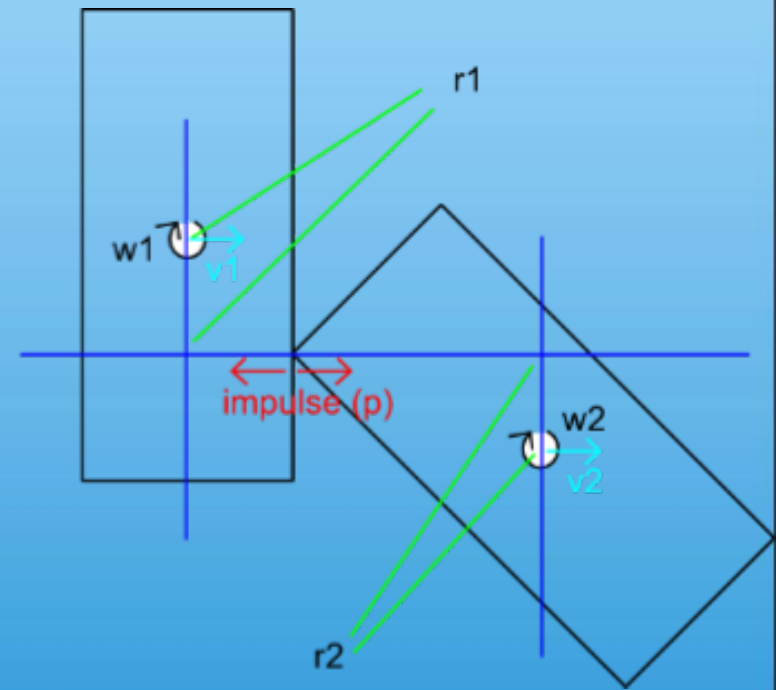
# Impulses

- Impulse : a force applied instantaneously, measured in kg m/s
- Can change a shape's linear and/or angular velocity.
- Rotation depends on moment arm ( $r$ ).
- $\Delta v = \Delta p/m$ ,  $\Delta \omega = r\Delta p/I$  ( $I$  = moment of inertia)



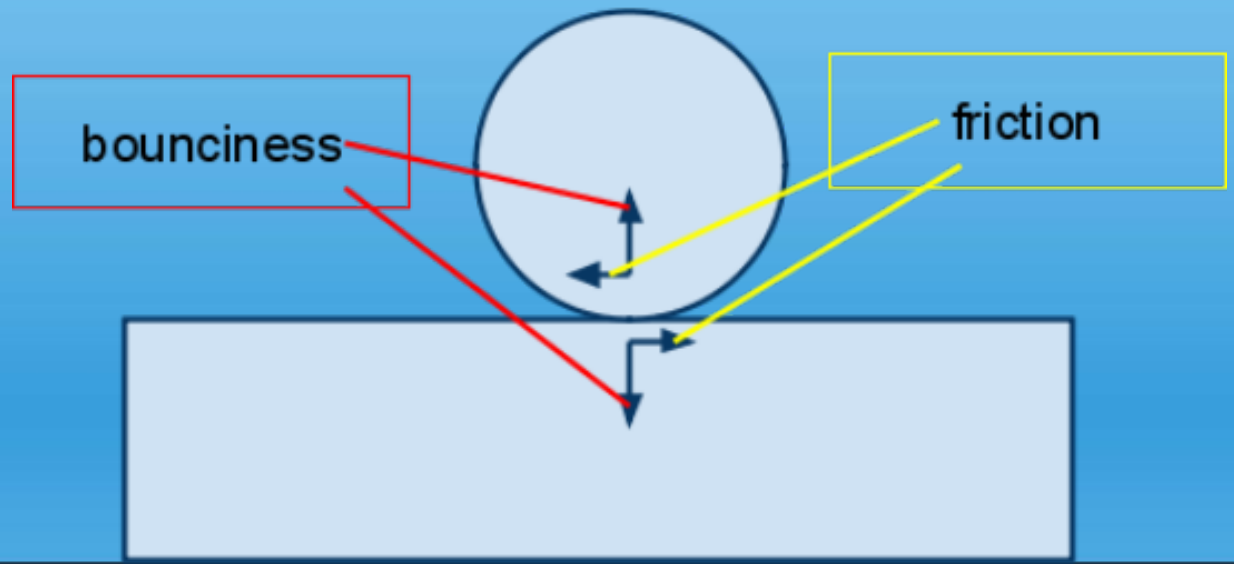
# Collisions

- Impulses push the objects apart
- Elastic collisions preserve kinetic energy
- $E_k = \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2$
- Both angular and linear velocity changes
- $$p = \frac{2(v_1 - v_2 + w_1 r_1 - w_2 r_2)}{(1/m_1 + 1/m_2 + r_1^2/I_1 + r_2^2/I_2)}$$

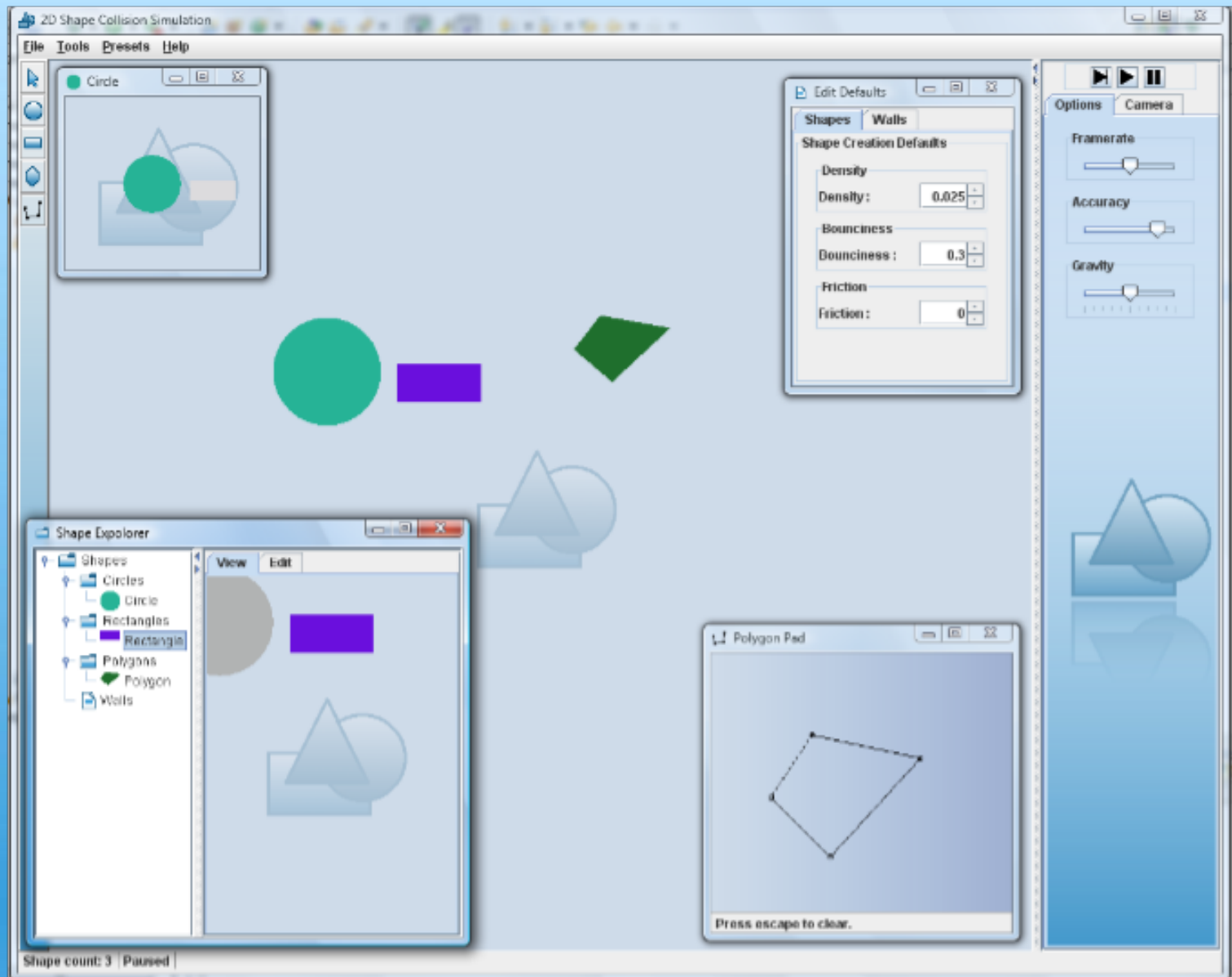


# More on Collisions

- Objects have friction and bounciness
- $p = \frac{2(v_1 - v_2 + w_1 r_1 - w_2 r_2)}{(1/m_1 + 1/m_2 + r_1^2/I_1 + r_2^2/I_2)}$
- This formula is only for elastic collisions
- For inelastic collisions, the 2 is replaced by  $1 + b_1 b_2$  (the bounciness factors,  $0 \leq b \leq 1$ )
- Friction:  $\mu = f_1 f_2$  (the friction factors,  $0 \leq f \leq 1$ )
- Friction impulse =  $\mu * p$  (normal impulse)

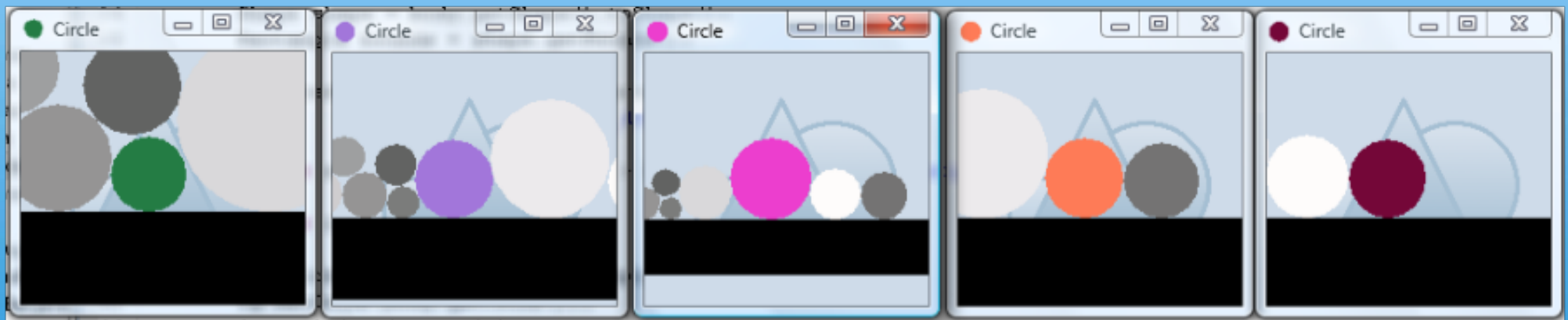


# GUI



# Structure

- Modular and extendable
- Model / view separation
  - GUI in different package from model
  - Model: the actual shapes and math
  - GUI: the user interface, uses the model





# Synchronization

- Concurrency
  - Animation always requires multiple threads
- Manual synchronization
  - The two main lists needed to be synchronized manually

```
145 protected void step(double amount) {  
146     // collide every body with every other body  
147     // but don't do reverse collisions  
148     // e.g. don't do both b.collide(a) and a.collide(b)  
149     synchronized (bodies) {  
150         for (int i = 0; i < bodies.size(); ++i) {  
151             Body b1 = bodies.get(i);  
152             for (int j = i + 1; j < bodies.size(); ++j) {  
153                 Body b2 = bodies.get(j);  
154                 b1.collide(b2);  
155             }  
156         }  
157     }  
158  
159     synchronized (bodies) {  
160         // step all bodies  
161         for (Body b : bodies) {  
162             b.step(amount);  
163             // gravity  
164             b.addImpulse(new Vector(0, 1 * amount * b.mass() * gravity));  
165         }  
166     }  
167  
168     synchronized (springs) {  
169         // step all springs  
170         for (Spring s : springs) {  
171             s.step(amount);  
172         }  
173     }  
174  
175     synchronized (grablock) {  
176         if (grabbed != null) {  
177             Vector mouse = new Vector(mouseX, mouseY);  
178             Vector diff = mouse.subtract(grabbed.connectedPoint());  
179             Vector result = diff.multiply(10 * amount);  
180             grabbed.addImpulse(result);  
181             grabbed  
182                 .addImpulse(grabbed.getVelocity())  
183                 .multiply(-10 * amount);  
184         }  
185     }  
186 }
```

# Documentation

- Javadocs (auto-generated html-based documentation for Java)
- Could help other people working on similar projects

The screenshot displays the Javadoc page for the `simulation.gui.MainWindow` class. The page is organized into several sections:

- Navigation:** On the left, there are links for "All Classes", "Packages" (including `simulation.engine` and `simulation.gui`), and "All Classes" (listing various classes like `AboutWindow`, `AnimationClass`, etc.).
- Class Hierarchy:** A tree diagram shows the inheritance path: `java.lang.Object` → `java.awt.Component` → `java.awt.Container` → `java.awt.Window` → `java.awt.Frame` → `javax.swing.JFrame` → `simulation.gui.MainWindow`.
- All Implemented Interfaces:** Lists interfaces implemented by `MainWindow`, including `java.awt.image.ImageObserver`, `java.awt.MenuContainer`, `java.io.Serializable`, `javax.accessibility.Accessible`, `javax.swing.RootPaneContainer`, and `javax.swing.WindowConstants`.
- Class Declaration:** Shows the public class declaration: `public class MainWindow extends javax.swing.JFrame`.
- Description:** A paragraph explaining that this is the main window of the simulation, containing the main display and the model.
- Author:** Tikhon Jelvis.
- See Also:** A link to the `Serialized Form`.
- Nested Class Summary:** A section header for nested classes and interfaces.
- Nested classes/interfaces inherited from class java.awt.Component:** A list of nested classes and interfaces inherited from `java.awt.Component`.
- Field Summary:** A section header for field summaries.
- Field Summary Table:** A table with two columns: "Field" and "Description". It lists fields like `RESOURCE_PATH` and `VERSION` with their descriptions.
- Fields inherited from class javax.swing.JFrame:** A list of fields inherited from `javax.swing.JFrame`.
- Fields inherited from class java.awt.Frame:** A list of fields inherited from `java.awt.Frame`.
- Fields inherited from class java.awt.Component:** A list of fields inherited from `java.awt.Component`.