Mission: determine if there is a collision between asteroid and satellite.

\[
\vec{p}^{k+1} = \vec{p}^k + \Delta t \; \vec{v}(\vec{p}^k, t^k).
\]

- **satellite**: \( \vec{v}_s(\vec{p}, t) = (-y, x) \) where \( \vec{p} = (x, y) \), starts at \( \vec{p}_s = (25, 0) \).
- **asteroid**: \( \vec{v}_a(\vec{p}, t) = (-30, -2.5) \), starts at \( \vec{p}_a = (50, 25) \).
- use \( \Delta t = 0.2 \).
- Collision if distance is less than 2 (in provided units).
- Should we fire thrusters to avoid asteroid? (fuel is expensive!)

<table>
<thead>
<tr>
<th>t</th>
<th>ps</th>
<th>vs(ps, t)</th>
<th>pa</th>
<th>va(pa, t)</th>
<th>distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(25, 0)</td>
<td>(-0, 25)</td>
<td>(50, 25)</td>
<td>(-30, -2.5)</td>
<td>35.4</td>
</tr>
<tr>
<td>0.2</td>
<td>(25, 5)</td>
<td>(-5, 25)</td>
<td>(44, 24.5)</td>
<td>(-30, -2.5)</td>
<td>27.2</td>
</tr>
<tr>
<td>0.4</td>
<td>(24, 10)</td>
<td>(-10, 24)</td>
<td>(38, 24)</td>
<td>(-30, -2.5)</td>
<td>19.8</td>
</tr>
</tbody>
</table>
Revisiting the Pixar ball from last lecture & lab.

- forces act on objects
- change in motion over some time is proportional to sum of forces acting on object.

\[ \ddot{a} = \Delta \frac{\Delta \vec{v}}{\Delta t} \]

\[ \sum \vec{F} = m \ddot{g} \]

\[ \vec{g} = \begin{pmatrix} 0 \\ -9.81 \end{pmatrix} \]

Input: initial position \( \vec{p}_0 \), velocity \( \vec{v}_0 \)

Output: \( \vec{p}_t \), \( \vec{v}_t \)

Analytic solution:

\[ y = y_0 + v_{0y} t - \frac{1}{2} g t^2 \]
Things get more complicated with other forces.

Add drag

analytic solution ???

\[ f_d = -\frac{1}{2} \rho A v^2 \text{area} \]

\[ \text{density of air} \]

\[ \text{o.s. for a sphere} \]

\[ \vec{F}_g = mg \]

\[ \vec{g} = \begin{pmatrix} 0 \\ -9.81 \\ 0 \end{pmatrix} \]

\[ y = 0 \]
Let's just use a computer...

\[ \vec{v}^k = \frac{\Delta p}{\Delta t} = \frac{\vec{p}^{k+1} - \vec{p}^k}{\Delta t} \]

\[ \vec{a}^k = \frac{\Delta v}{\Delta t} = \frac{\vec{v}^{k+1} - \vec{v}^k}{\Delta t} \]

\[ \vec{p}^{k+1} = \vec{p}^k + \Delta t \vec{v}^k \]

\[ \vec{v}^{k+1} = \vec{v}^k + \Delta t \vec{a}^k \]

\[ m \vec{a} = \sum f \]

\[ \vec{a} = \frac{1}{m} \sum \vec{f} \]

\( k \) is the iteration (frame) in animation.
Draw particles with `gl.drawArrays` and `gl.POINTS`.

**JS**

```javascript
1. gl.bindBuffer(gl.ARRAY_BUFFER, positionBuffer);
2. gl.vertexAttribPointer(a.Position, 3, gl.FLOAT, false, 0, 0);
3. gl.drawArrays(gl.POINTS, 0, nVertices);
```

**VS**

```plaintext
attribute vec4 a_Position;

uniform mat4 u_ProjectionMatrix;
uniform mat4 u_ViewMatrix;

void main()
{
    gl_Position = u_ProjectionMatrix * u_ViewMatrix * vec4(a_Position, 1);
    gl_PointSize = 50.0 / gl_Position.w;
}
```

**FS**

```plaintext
uniform sampler2D tex_Sprite;

void main()
{
    gl_FragColor = texture2D(tex_Sprite, gl_PointCoord);
    // gl_FragColor = vec4(1, 1, 1, 1);
    // gl_FragColor = vec4(1, 1, 1, 1);
}
```
We'll use *transform feedback* to capture updated position & velocity to a buffer and send to next iteration.

- **Buffer for current position**: $p^k$
- **Buffer for updated position**: $p^{k+1}$

**Initialize with particle positions.**

Swap at the end of iteration.

Apply equations from Euler's method:

$$\frac{\Delta p}{\Delta t}^{k+1} = p^k + \Delta t \cdot v^k$$

In vertex shader, capture this in a varying.