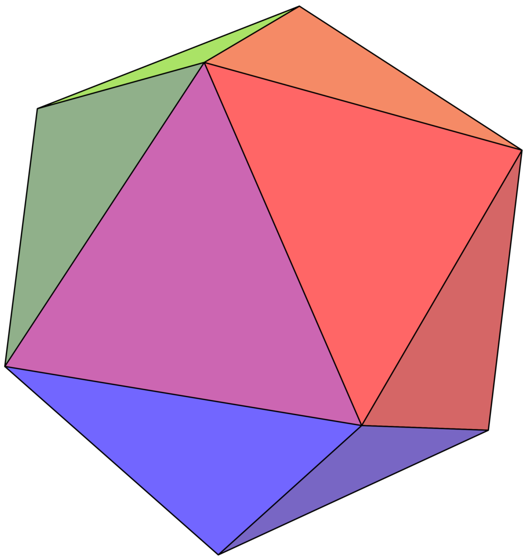
Physics3D

Lennart Van Hirtum

February 2021



1 Shapes

1.1 Fundamental formulae

Volume

$$V = \iiint_V 1 \; dxdydz \tag{1}$$

Center of mass

$$\vec{M} = \frac{\iiint_V \left(\begin{smallmatrix} x \\ y \\ z \end{smallmatrix}\right) dxdydz}{V} \tag{2}$$

Moment of inertia

$$\hat{I} = \iiint_V [\left(\begin{smallmatrix} x \\ y \\ z \end{smallmatrix}\right)]^2_\times dxdydz = \iiint_V \left(\begin{smallmatrix} -y^2-z^2 & xy & xz \\ xy & -x^2-z^2 & yz \\ xz & yz & -x^2-y^2 \end{smallmatrix}\right) dxdydz \tag{3}$$

1.2 Shape Classes

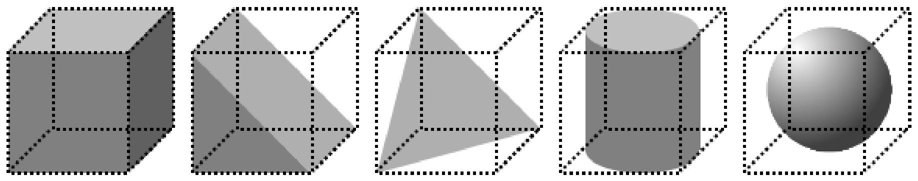


Figure 1: Shape classes, left to right: Box, Wedge, Corner, Cylinder, Sphere

Shape classes are a way of solving two problems at once:

1. Not all shapes are representable by polyhedra (ex, wheels, balls, toruses).
2. A Polyhedron uses a lot of memory - especially when trying to approximate these non-representable shapes - much of which can be saved if multiple parts could reuse the same polyhedron.

We do this by separating the 'shape type' and the size of the object. This 'shape type' then describes the shape in an abstract manner, it can be a polyhedron, but it can also be a cylinder, a sphere, or any other shape, such as Box, Wedge and Corner to optimize common cases. Instead of a plain Polyhedron, a Shape now has a pointer to an existing ShapeClass, and a scale factor[1] (s_x, s_y, s_z). A ShapeClass is defined to have bounds -1..1 in all axes, these bounds are then rescaled to the size of the object by the scale factor. This way a single polyhedron can be the basis for many objects.

[1]Not all shapes can be scaled freely. Spheres and cylinders require that the certain proportions are kept so that all circles remain circles. This is enforced by making the ShapeClass responsible for applying scaling factors to a Shape

1.2.1 Scaling

Scaling a shape along the major axes with s_x, s_y, s_z Volume and Center of Mass are trivial to scale:

$$V' = s_x s_y s_z V \tag{4}$$

$$\vec{M}' = \begin{pmatrix} s_x M_x \\ s_y M_y \\ s_z M_z \end{pmatrix} \tag{5}$$

But the inertial matrix is another story. We will assume a shorthand notation to build the full integral

$$\begin{aligned}
I_{xy} &= \iiint_V xy \ dxdydz \\
I_{xz} &= \iiint_V xz \ dxdydz \\
I_{yz} &= \iiint_V yz \ dxdydz \\
I_{x^2} &= \iiint_V x^2 \ dxdydz \\
I_{y^2} &= \iiint_V y^2 \ dxdydz \\
I_{z^2} &= \iiint_V z^2 \ dxdydz
\end{aligned} \tag{6}$$

Which makes the original inertia matrix

$$\hat{I} = \begin{pmatrix} -I_{y^2} - I_{z^2} & I_{xy} & I_{xz} \\ I_{xy} & -I_{x^2} - I_{z^2} & I_{yz} \\ I_{xz} & I_{yz} & -I_{x^2} - I_{y^2} \end{pmatrix} \tag{7}$$

For each of these terms we can compute a scaled version:

$$\begin{aligned}
I'_{xy} &= \iiint_V s_x x s_y y \ s_x dx s_y dy s_z dz = s_x^2 s_y^2 s_z I_{xy} \\
I'_{x^2} &= \iiint_V (s_x x)^2 \ s_x dx s_y dy s_z dz = s_x^3 s_y s_z I_{x^2}
\end{aligned} \tag{8}$$

likewise for $I_{xz}, I_{yz}, I_{y^2}, I_{z^2}$.
This makes the resulting scaled inertial matrix:

$$\hat{I}' = s_x s_y s_z \begin{pmatrix} -s_y^2 I_{y^2} - s_z^2 I_{z^2} & s_x s_y I_{xy} & s_x s_z I_{xz} \\ s_x s_y I_{xy} & -s_x^2 I_{x^2} - s_z^2 I_{z^2} & s_y s_z I_{yz} \\ s_x s_z I_{xz} & s_y s_z I_{yz} & -s_x^2 I_{x^2} - s_y^2 I_{y^2} \end{pmatrix} \tag{9}$$

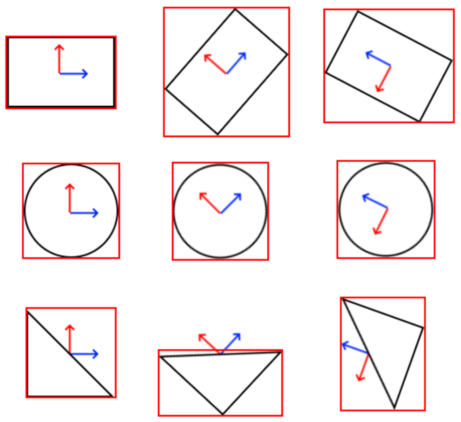


Figure 2: Local bounds of rotated shape classes

1.2.2 Global Bounds Computation

To compute the global bounds of an object, we need some general way of computing bounds of ShapeClasses. Translating a shape and it's bounds is easy, just apply the same translation. Rotation and scaling however, do affect the fundamental shape of the bounds, and so must be handled by ShapeClass. This is done by `BoundingBox getBounds(Rotation rotation, DiagonalMat3 scale)`.

- Sphere: The bounds for a sphere are quite simple, just $-r..r$ in all directions. (with $r = s_x = s_y = s_z$ due to the scaling restriction of sphere)
- Cylinder: We will define a cylinder along the z-axis. So $r = s_x = s_y$. TODO bounds
- Box: By We can take the base vectors of the given rotation, and write out the matrix products to project each of the corners of the box:

$$\hat{R}\begin{pmatrix} \pm s_x \\ \pm s_y \\ \pm s_z \end{pmatrix} = \pm s_x \vec{R}_0 \pm s_y \vec{R}_1 \pm s_z \vec{R}_2 \tag{10}$$

  With $\vec{R}_0, \vec{R}_1, \vec{R}_2$ the columns of rotation matrix $\hat{R}$. The 8 possible choices for the $\pm$ describe the 8 corners of the box. s_x, s_y, s_z represent the scales in each direction of the box

1.3 Polyhedra

We need to compute these properties for polyhedra as well. Since polyhedra in Physics3D are defined by their surface, we will convert the volume intervals to surface intervals using Gauss' Theorem[2]. Using $\vec{n} = (\vec{v_1} - \vec{v_0}) \times (\vec{v_2} - \vec{v_0})$, $\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \circ \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \begin{pmatrix} a_x b_x \\ a_y b_y \\ a_z b_z \end{pmatrix}$, and $\vec{v}^2 = \vec{v} \circ \vec{v}$

Volume:

$$V = \iiint_V 1 \; dV = \oiint_S \begin{pmatrix} 0 \\ 0 \\ z \end{pmatrix} \cdot \vec{n} \; dS \tag{11}$$

$$= \sum_{t=\{\vec{v_0},\vec{v_1},\vec{v_2}\}}^{Triangles} \frac{n_z v_{0z} v_{1z} v_{2z}}{6} \tag{12}$$

Center Of Mass:

$$\vec{M} = \frac{\iiint_V \begin{pmatrix} x \\ y \\ z \end{pmatrix} dV}{V} = \frac{\oiint_S \begin{pmatrix} x^2/2 \\ y^2/2 \\ z^2/2 \end{pmatrix} \cdot \vec{n} \; dS}{V} \tag{13}$$

$$= \sum_{t=\{\vec{v_0},\vec{v_1},\vec{v_2}\}}^{Triangles} \frac{\vec{v_0}^2 + \vec{v_1}^2 + \vec{v_2}^2 + \vec{v_0} \circ \vec{v_1} + \vec{v_1} \circ \vec{v_2} + \vec{v_2} \circ \vec{v_0}) \circ \vec{n}}{24V} \tag{14}$$

Moment of inertia:

$$I_{xy} = \iiint_V xy \; dV = \oiint_S \begin{pmatrix} 0 \\ 0 \\ xyz \end{pmatrix} \cdot \vec{n} \; dS \tag{15}$$

$$I_{x^2} = \iiint_V -y^2 - z^2 \; dV = \oiint_S \begin{pmatrix} 0 \\ -y^3/3 \\ -z^3/3 \end{pmatrix} \cdot \vec{n} \; dS \tag{16}$$

likewise for $I_{xz}, I_{yz}, I_{y^2}, I_{z^2}$. The concrete triangles solution is quite long, an implementation can be found in the source code[3].

[2] `https://en.wikipedia.org/wiki/Divergence_theorem`
[3] polyhedron.cpp

2 Constraints

The equation including other constraints' effects is:

$$\hat{ME}_a * \left(\begin{pmatrix} \vec{v_{a_0}} \\ \vec{w_{a_0}} \end{pmatrix} + \sum_i^{c_a} \hat{PM}_i \vec{p_i} \right) = \hat{ME}_b * \left(\begin{pmatrix} \vec{v_{b_0}} \\ \vec{w_{b_0}} \end{pmatrix} + \sum_i^{c_b} \hat{PM}_i \vec{p_i} \right) \tag{17}$$

With c_a (c_b) the set of constraints that link to the first (second) part of the constraint. $\hat{ME}_a$ the motion-to-equation matrix. $\hat{PM}_i$ the parameter-to-motion matrices for the other constraints linking to the included objects. $\vec{p_i}$ the parameter vector for each involved constraint.

Final result:

$$\hat{ME}_a * \sum_i^{c_a} \hat{PM}_i \vec{p_i} - \hat{ME}_b * \sum_i^{c_b} \hat{PM}_i \vec{p_i} = Err \tag{18}$$

$$Err = \hat{ME}_b * \begin{pmatrix} \vec{v_{b_0}} \\ \vec{w_{b_0}} \end{pmatrix} - \hat{ME}_a * \begin{pmatrix} \vec{v_{a_0}} \\ \vec{w_{a_0}} \end{pmatrix} \tag{19}$$

2.1 Formula

$$\begin{bmatrix} ME_{a1} * PM_{a_1 1} & \cdots & ME_{a1} * PM_{a_1 k} \\ -ME_{b1} * PM_{b_1 1} & & -ME_{b1} * PM_{b_1 k} \\ \vdots & \ddots & \vdots \\ ME_{aj} * PM_{a_j 1} & \cdots & ME_{aj} * PM_{a_j k} \\ -ME_{bj} * PM_{b_j 1} & & -ME_{bj} * PM_{b_j k} \end{bmatrix} \begin{pmatrix} \overrightarrow{P_1} \\ \vdots \\ \overrightarrow{P_k} \end{pmatrix} = \begin{pmatrix} ME_{b1} * \overrightarrow{M_{b_1 0}} \\ -ME_{a1} * \overrightarrow{M_{a_1 0}} \\ \vdots \\ ME_{bj} * \overrightarrow{M_{b_j 0}} \\ -ME_{aj} * \overrightarrow{M_{a_j 0}} \end{pmatrix} \tag{20}$$

1. $PM_{a_j} k$ is the effect of the parameters of constraint k on the motion of object attached to the a side of constraint j

2. ME_{aj} is the Motion to Equation Matrix. It converts the motion of side a of the constraint to the values which can be summed to get the full constraint equation. We might want to assert that the motions are equal, but this is too strict, for example, for a ballconstraint, only the velocities need to be equal, not the angular velocities

2.2 Derivation for BallConstraint

```
struct BallConstraint {
    Vec3 attach1; //r_a
    Vec3 attach2; //r_b
}
```

For BallConstraint, the equation we wish to satisfy is the following:

$$\vec{v_a} + \vec{r_a} \times \vec{w_a} = \vec{v_b} + \vec{r_b} \times \vec{w_b} \tag{21}$$

This gives rise to the following matrices, applied to the motion vector $\left(\begin{smallmatrix}\vec{v}\\ \vec{w}\end{smallmatrix}\right)$

$$\left[\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) \quad [\vec{r_a}]_\times\right]\begin{pmatrix}\vec{v_a}\\ \vec{w_a}\end{pmatrix} = \left[\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) \quad [\vec{r_b}]_\times\right]\begin{pmatrix}\vec{v_b}\\ \vec{w_b}\end{pmatrix} \tag{22}$$

The forces of a BallConstraint on the motion of it's connected objects must also be incorporated:

$$\Delta\vec{v} = m^{-1}\vec{p} \tag{23}$$

$$\Delta\vec{w} = [\hat{I}^{-1}](\vec{r} \times \vec{p}) \tag{24}$$

Convert to a matrix:

$$\Delta\begin{pmatrix}\vec{v}\\ \vec{w}\end{pmatrix} = \begin{bmatrix} m^{-1}\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right)\\ \hat{I}^{-1}[\vec{r}]_\times \end{bmatrix}\vec{p} \tag{25}$$

Resulting Matrices:

$$P\hat{M}_a = \begin{bmatrix} m_a^{-1}\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right)\\ \hat{I}_a^{-1}[\vec{r_a}]_\times \end{bmatrix} \tag{26}$$

$$P\hat{M}_b = \begin{bmatrix} m_b^{-1}\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right)\\ \hat{I}_b^{-1}[\vec{r_b}]_\times \end{bmatrix} \tag{27}$$

$$M\hat{E}_a = \left[\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) \quad [\vec{r_a}]_\times\right] \tag{28}$$

$$M\hat{E}_b = \left[\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) \quad [\vec{r_b}]_\times\right] \tag{29}$$

2.3 Derivation for HingeConstraint

```
struct HingeConstraint {
    Vec3 attach1; // r_a
    Vec3 axis1;
    Vec3 attach2; // r_b
    Vec3 axis2;
}
```

We will assume that the constraint is satisfied in position, so both axes are parallel
Let $\vec{z}$ be the global axis of rotation for the hinge, with $\vec{x}$ and $\vec{y}$ such that $\vec{x}\perp\vec{y}\perp\vec{z}\perp\vec{x}$

The equations that must be satisfied are:

$$\vec{v_a} + \vec{r_a} \times \vec{w_a} = \vec{v_b} + \vec{r_b} \times \vec{w_b} \tag{30}$$

$$(\vec{w_a} - \vec{w_b}) \cdot \vec{x} = 0 \tag{31}$$

$$(\vec{w_a} - \vec{w_b}) \cdot \vec{y} = 0 \tag{32}$$

Possible Forces:
Hinges can exert forces/impulses in any direction other than rotating around the axis of rotation

$$\Delta\vec{v} = m^{-1}\vec{p_t} \tag{33}$$

$$\Delta\vec{w} = [\hat{I}^{-1}](\vec{r} \times \vec{p_t} + p_x * \vec{x} + p_y * \vec{y}) \tag{34}$$

Resulting matrices:

$$P\hat{M}_a = \begin{bmatrix} m_a^{-1}\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) & \left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right) & \left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right) \\ \hat{I}_a^{-1}[\vec{r_a}]_\times & \hat{I}^{-1}\vec{x} & \hat{I}^{-1}\vec{y} \end{bmatrix} \tag{35}$$

$$P\hat{M}_b = \begin{bmatrix} m_b^{-1}\left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) & \left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right) & \left(\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right) \\ \hat{I}_b^{-1}[\vec{r_b}]_\times & \hat{I}^{-1}\vec{x} & \hat{I}^{-1}\vec{y} \end{bmatrix} \tag{36}$$

$$M\hat{E}_a = \begin{bmatrix} \left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) & [\vec{r_a}]_\times \\ \left(\begin{smallmatrix}0&0&0\end{smallmatrix}\right) & \vec{x}^T \\ \left(\begin{smallmatrix}0&0&0\end{smallmatrix}\right) & \vec{y}^T \end{bmatrix} \tag{37}$$

$$M\hat{E}_b = \begin{bmatrix} \left(\begin{smallmatrix}1&0&0\\0&1&0\\0&0&1\end{smallmatrix}\right) & [\vec{r_b}]_\times \\ \left(\begin{smallmatrix}0&0&0\end{smallmatrix}\right) & \vec{x}^T \\ \left(\begin{smallmatrix}0&0&0\end{smallmatrix}\right) & \vec{y}^T \end{bmatrix} \tag{38}$$

2.4 Derivation for BarConstraint

```
struct barConstraint {
    Vec3 attach1; // r_a
    Vec3 attach2; // r_b
    double length;
}
```

Let d be the vector between the points
The equation that must be satisfied is:

$$(\vec{v_a} + \vec{r_a} \times \vec{w_a}) \cdot \vec{d} = (\vec{v_b} + \vec{r_b} \times \vec{w_b}) \cdot \vec{d} \tag{39}$$

Possible Forces:
A bar can only excert forces/impulses along the length of the bar:

$$\Delta\vec{v} = m^{-1} * \vec{d} * p \tag{40}$$

$$\Delta\vec{w} = [\hat{I}^{-1}](\vec{r} \times \vec{d}) * p \tag{41}$$

Resulting matrices:

$$P\hat{M}_a = \begin{bmatrix} m_a^{-1}\vec{d} \\ \hat{I}_a^{-1}(\vec{r_a} \times \vec{d}) \end{bmatrix} \tag{42}$$

$$P\hat{M}_b = \begin{bmatrix} m_b^{-1}\vec{d} \\ \hat{I}_b^{-1}(\vec{r_b} \times \vec{d}) \end{bmatrix} \tag{43}$$

$$M\hat{E}_a = \begin{bmatrix} \vec{d} & \vec{r_a} \times \vec{d} \end{bmatrix} \tag{44}$$

$$M\hat{E}_b = \begin{bmatrix} \vec{d} & \vec{r_b} \times \vec{d} \end{bmatrix} \tag{45}$$