Collision Detection

Half Spaces

A half space H is characterized by a pair $H = (\overline{n}, d)$, where $\overline{n}$ is the normal vector, and d is the displacement.

If $\overline{x}$ is a point, and $H = (\overline{x}, d)$ is a half space, then the distance of $\overline{x}$ from H, written as $\Delta(\overline{x}, H)$, is defined as

$$\Delta(\overline{x}, H) = \overline{x} \cdot \overline{n} - d.$$

Half Spaces

Alternatively, a half space can be defined by its (outside pointing) normal vector $\overline{n}$, and a point $\overline{b}$ on the border. Then

$$H = (\ \overline{n},\ \ \overline{n} \cdot \overline{b}\ )$$

has $\Delta(\overline{b}, H) = 0$, so that $\overline{b}$ is on really the border.

If $\overline{n}$ is a unit vector and $\overline{b}$ a point, then $\Delta(\overline{x}, (\ \overline{n},\ \ \overline{n} \cdot \overline{b}\ )\ )$ is the distance between $\overline{x}$ and the border of the halfspace defined by normal vector $\overline{n}$ and some border point $\overline{b}$.

We say that $\overline{x}$ is inside the half space H if $\Delta(\overline{x}, H) \leq 0$.

Other Forms

One could consider using other forms as well, for example spheres or cylinders.

A sphere could be defined by a center point $\overline{c}$ and a distance d.

For a sphere $S = (\overline{c}, d)$, the distance $\Delta(\overline{x}, S)$ would be defined as $\|\overline{x} - \overline{c}\| - d$.

This may be a good idea, but we will not further follow this.

Boolean Expressions

Using halfspaces (and possible other basic components), we can define shapes as arbitrary (positive) Boolean expressions over half spaces.

- A halfspace H is a shape.
- $\perp$ is a shape.
- $\top$ is a shape.
- If $S_1, \ldots, S_n$ are shapes, then $S_1 \cup \cdots \cup S_n$ and $S_1 \cap \cdots \cap S_n$ are shapes.

(It seems OK to assume that shapes can be three-dimensional in English language.)

House

Consider a (very simple) house defined by points

$$(3,4,0),\ (-3,4,0),\ (-3,-4,0),\ (-3,4,0),\ (0,4,7),\ (0,-4,7).$$

Its shape can be defined by:

$$S = \bigcap \left\{ \begin{array}{ll} H(\ (0,-1,0),\ 4\ ) & \text{wall at } Y=-4 \\ H(\ (0,1,0),\ 4\ ) & \text{wall at } Y=4 \\ H(\ (1,0,0),\ 3\ ) & \text{wall at } X=3 \\ H(\ (-1,0,0),\ 3\ ) & \text{wall at } X=-3 \\ H(\ (1,0,1),\ 7\ ) & \text{roof} \\ H(\ (-1,0,1),\ 7\ ) & \text{roof} \end{array} \right.$$

Some of the normal vectors are not of unit length, but this is not important.

Checking whether a point is inside a shape S is easy:

- $\overline{x} \in (\overline{n}, d)$ if $\Delta(\ \overline{x},\ \ S\ ) \leq 0$.
- $\overline{x} \in \bigcup H_i$ if there is an i, s.t. $\overline{x} \in H_i$.
- $\overline{x} \in \bigcap H_i$ if for all i, we have $\overline{x} \in H_i$.
- Always $\overline{x} \in \top$.
- Never $\overline{x} \in \perp$.

But nearly always, the real problem that one wants to solve is the question of collisions:

Given two points $(\overline{x}_1, \overline{x}_2)$, find smallest $\lambda \in [0, 1]$, for which $\overline{x}_1 + \lambda(\overline{x}_2 - \overline{x}_1)$ in S.

Definition: Let S be a shape, let $\overline{x}_1$ and $\overline{x}_2$ be points. Let $\lambda_1 \leq \lambda_2$ be in $\mathcal{R}$. The entry point $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S)$ is defined as follows:

- If there exists a $\lambda \in [\lambda_1, \lambda_2]$, s.t. $\overline{x}_1 + \lambda(\overline{x}_2 - \overline{x}_1) \in S$, then choose λ in such a way that there is no $\lambda' \in [\lambda_1, \lambda\rangle$ with $\overline{x}_1 + \lambda'(\overline{x}_2 - \overline{x}_1) \in S$.
  - If $\overline{x}_1 + \lambda(\overline{x}_2 - \overline{x}_1)$ lies on a border $(\overline{n}, d)$ of S, then
  $$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda, \overline{n}).$$
  - Otherwise,
  $$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda, \overline{0}).$$
- If no such λ exists, then
$$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda', \overline{0}), \text{ for some } \lambda' > \lambda_2.$$

Algorithm for $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S)$

- If S is a halfspace $H = (\overline{n}, d)$, then first compute

$$\begin{cases} \mu_1 & = & \Delta(\overline{x}_1 + \lambda_1(\overline{x}_2 - \overline{x}_1), H) \\ \mu_2 & = & \Delta(\overline{x}_1 + \lambda_2(\overline{x}_2 - \overline{x}_1), H) \end{cases}$$

If $\mu_1 < 0$, then $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, H) = (\lambda_1, \overline{0})$.

If $\mu_1 = 0$, then $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, H) = (\lambda_1, \overline{n})$.

If $\mu_1 > 0$, and $\mu_2 \leq 0$, then

$$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, H) = (\lambda_1 - \mu_1 \frac{\lambda_2 - \lambda_1}{\mu_2 - \mu_1}, \overline{n}).$$

If $\mu_1 > 0$ and $\mu_2 > 0$, then $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, H) = (\lambda_2 + 1000, \overline{0})$.

Algorithm for $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S)$ (2)

- If S has form $\bigcup_{i=1}^{n} S_i$, then start by setting $(\lambda, \overline{n}) = (\lambda_2 + 1000, \overline{0})$.

  For each i with $1 \leq i \leq n$, do the following:

  - If $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \min(\lambda, \lambda_2), S_i) = (\lambda', \overline{n}')$ and $\lambda' < \lambda$, then replace $(\lambda, \overline{n})$ by $(\lambda', \overline{n}')$.

  When the loop is complete, $(\lambda, \overline{n})$ equals

$$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S).$$

Algorithm for $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S)$ (3)

- If S has form $\bigcap_{i=1}^{n} S_i$, then start by setting $(\lambda, \overline{n}) = (\lambda_1, \overline{0})$.

  As long as $\lambda \leq \lambda_2$ and there exists an S_i with $1 \leq i \leq n$, for which $\Phi(\overline{x}_1, \overline{x}_2, \lambda, \lambda_2, S_i) = (\lambda', \overline{n}')$, and either

  1. $\lambda' > \lambda$ or
  2. $\lambda' = \lambda, \quad \|\overline{n}'\| \neq 0$ and $\|\overline{n}\| = 0$,

  replace $(\lambda, \overline{n})$ by $(\lambda', \overline{n}')$.

  When no further replacements are possible, we have computed

$$\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda, \overline{n}).$$

In case each S_i is a halfspace, it is sufficient to use a single for loop.

Algorithm for $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S)$ (4)

- If $S = \perp$, then $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda_2 + 1000, \overline{0})$.
- If $S = \top$, then $\Phi(\overline{x}_1, \overline{x}_2, \lambda_1, \lambda_2, S) = (\lambda_1, \overline{0})$.

Material Properties

We now whether we are touching (or colliding with something), but we also need to know the material properties of what we are touching. It seems reasonable to remember the following properties:

- A static friction coefficient μ_s.
- A dynamic friction coefficient μ_k.
- Maximum force F_m. This is the maximal force that the material can withstand before it breaks or gives way. For water, it would be 0, for sand or grass it would be some low number. For concrete, it should be a number that is high enough to carry the plane.

Adding Material Properties

It is not clear where material properties should be stored. I can think of the following possibilities:

1. Store properties in every halfspace H.

2. Introduce a material operator $M(P, S)$ (in addition to the Boolean operators) denoting: Every halfspace in S is made of material P.

3. In addition to **(2)**, add a modification operator to the Boolean expressions: $M(S_1, P, S_2)$ denoting: Whenever the contact point in inside S_1 (but not necessarily on the border of S_1), it is made of P. Otherwise, its material is termined by S_2.

Operator **(3)** makes it possible to 'paint' on the surface of S_2. For example S_1 specifies a road pattern, S_2 specifies the landscape.

Modeling Wheels

Friction

Friction coefficients are defined by the two materials that touch each other. Since we cannot store friction coefficients for all possible combinations of materials, we store them only for the rubber that our wheels are made of. We could also store friction coefficients for metal. (In case we want to model very rough landings, or landings with broken gear.)

Here are some reasonable values:

- Wet runway/wheels: 0.1
- Dry runway/wheels: 0.4.
- Icy runway/wheels: 0.001.
- Dirty runway/wheels: 0.3?

(Published numbers vary quite a lot. Use your intuition and hope for the best.)

A wheel can be in three states:

1. Not in contact with the ground. There is no friction.
2. In contact and slipping (or rolling). In this case, friction causes resistance. The force does not depend on the surface area, or the amount of speed. Direction of force is opposite to the direction of speed, which must be parallel to the contact surface. Strength of force depends on the force that presses the wheels against the ground (called normal force). If there is no force pressing the wheels against the ground, we are in state 1. Otherwise, the force equals μ_k times the normal force.
3. In contact and not slipping. In that case, the friction force magically has exactly the right strength to stop every movement. The normal force must press the objects together, and the parallel force must be less than μ_s times the normal force.

The transition between state 2 and state 3 is subtle. If the wheel is in state 3 and the friction becomes more than $\mu_s \overline{F}_n$, then it goes into state 2. It stays in state 2 until the friction $\overline{F} = \mu_k \overline{F}_n$ is able to stop the movement, after which it enters state 3 again.

State 3 is difficult to model, because you first need first to know the total force from other sources, in order to be able to annihilate the force.

If the plane has more than one wheel, there is a circular dependency, which requires to solve a system of equations.

It may be difficult to determine how the different wheels share the force. This may be important to know in certain situations, in order to determine which wheel starts slipping first.

Pebble-and-Spring Model

As soon as first contact appears, we enter state 3 by putting a pebble on the surface. We assume that the wheel frame is connected to this pebble by springs. We try to keep the pebble on the same place as long as possible. As soon as we have to move the pebble, we are in state 2.

Wheel Frame Coordinates

We assume that the wheel frame, on the point where it is connected to the plane has its own coordinate system, defined by $\overline{b}$ and q.

The quaternion q takes steering into account. The axes are defined as follows:

X: Forward in the rolling direction of the wheel.

Y: To the right, in the direction of the axis of the wheel.

Z: Down.

We assume that we are always able to compute $\overline{b}$ and q from the airplane position, and steering input.

In addition, we sometimes use $\overline{v}$, speed of the origin of the wheel frame.

Pebble Border

The pebble border is defined as the area around the origin of the XY-plane that characterizes the forces in the XY-plane that the pebble can resist without slipping or rolling. We always assume that our normal direction of rolling is along the X-axis. The pebble border consists of the intersection of two patterns:

1. A circle with radius μF_z, where F_z is the normal force, and μ is the static or dynamic friction coefficient of the surface and the tyre.
2. A band of points (x, y, z) determined by $-B \leq x \leq B$, where B is the braking strength, which is input by the user.

If we are braking very hard, then the pebble border is completely determined by case 1, which depends on the friction of the tyre.

If we are not braking at all, then the border is a line segment on the Y-axis, between $(0, -\mu F_z, 0)$ to $(0, \mu F_z, 0)$.

Contact of Wheel on Surface

We need to determine our orientation of the surface. If the user is not braking too hard, then the orientation determines the direction in which the wheel will roll.

We define a function $Q(\overline{n})$ that defines a quaternion q with the following properties:

- Its rotation f_q maps $(1,0,0)$ into the rolling direction of the wheel over the surface. $f_q(1,0,0)$ defines the X-axis of the pebble border.
- f_q maps $(0,1,0)$ into the surface, into the direction that is orthogonal to $f_q(1,0,0)$. It corresponds to the Y-axis of the pebble border.
- f_q maps $(0,0,1)$ to a vector that is orthogonal to the surface (It will be parallel to $\overline{n}$.)

Orientation on the Surface (2)

The normal vector $\overline{n}$ must be in wheel frame coordinates. The result of $Q(\overline{n})$ is also in wheel frame coordinates. $Q(\overline{n})$ is not meaningful if $\dfrac{-\overline{n}_z}{\sqrt{\overline{n}_x^2 + \overline{n}_y^2}}$ is negative or small.

Define:

$$\phi_x = \arctan(\overline{n}_y, \sqrt{\overline{n}_x^2 + \overline{n}_z^2}\ ), \qquad \phi_y = \arctan(-\overline{n}_x, -\overline{n}_z).$$

Further define:

$$q_x = q_{(1,0,0),-\phi_x}, \qquad q_y = q_{(0,1,0),-\phi_y}.$$

(q_x is the rotation around $(1,0,0)$ over angle $-\phi_x$, and q_y is the rotation around $(0,1,0)$ over angle $-\phi_y$. Finally, $Q(\overline{n}) = q_y.q_x$.

Force from Pebble Position and Speed

We define a function $\overline{F}(\overline{p}, \overline{w})$, which computes the force that the pebble imposes on the wheel frame, assuming that it has position $\overline{p}$ and speed $\overline{w}$ in wheel frame coordinates.

The resulting force is also in wheel frame coordinates.

If $\overline{p}_z \geq z_{max}$, then the wheel is not in contact, and the force is $(0, 0, 0)$. (Remember that positive Z is downwards.)

If $\overline{p}_z \leq z_{min}$, then the wheel is too much compressed, and it breaks, in that case, one can either throw an exception, or change the wheel by some other model of a slipping plane.

(This would happen when the plane is overloaded, or after a hard landing.)

Force from Pebble Position and Speed (2)

We use the following parameters, which are all negative:

- k_{xy} is the horizontal spring coefficient, k_z is the vertical spring coefficient.
- d_{xy} is the horizontal damping coefficient, d_z is the vertical damping coefficient.
- z_{zero} is the zero position of the wheel, in which there is no vertical force.

$$\overline{F}(\overline{p}, \overline{w}) = \begin{cases} k_{xy}(\overline{p}_x, \overline{p}_y, 0) + k_z(0, 0, \overline{p}_z - z_{zero}) + \\ d_{xy}(\overline{w}_x, \overline{w}_y, 0) + d_z(0, 0, \overline{w}_z). \end{cases}$$

The force consists of the sum of the damping force and the elastic force.

Choice of Parameters

The values of the parameters must be chosen carefully. In physical reality, k_{xy} and d_{xy} are probably very high. Realistic values would probably result in numerical instability, so you will have to make them lower.

k_z and d_z probably can be given realistic values.

Clipping Force against Pebble Border

$\overline{F}_{cl}(\overline{F}, \mu, \overline{n})$ determines the maximal force that the pebble could impose on the wheel frame without slipping. Both $\overline{F}$ and $\overline{n}$ are in wheel frame coordinates. The result is also in wheel frame coordinates.

Define $q = Q(\overline{n})$, and define $\overline{F}' = f_{q^{-1}}(\overline{F})$. (This is $\overline{F}$, converted into coordinates based on the pebble border.) Let $\overline{F}'_{xy} = (\overline{F}'_x, \overline{F}'_y, 0)$ be the horizontal components of $\overline{F}'$.

If $|\overline{F}_x| \leq B$, and $\dfrac{\|\overline{F}'_{xy}\|}{F'_z} \leq \mu$, then $\overline{F}$ is inside the pebble border, so that $\overline{F}_{cl}(\overline{F}, \mu, \overline{n}) = \overline{F}$.

Otherwise, we are slipping or rolling, and $\overline{F}'$ has to be clipped against the pebble border. In that case, $\overline{F}_{cl} = f_q(\overline{H}_x, \overline{H}_y, \overline{F}'_z)$, where $\overline{H}$ is the horizontal force defined on the next slide, and $q = Q(\overline{n})$.

Clipping Force against Pebble Border (2)

Given a normal force $\overline{F}'_z$, and horizontal force $\overline{F}'_{xy}$ which is outside the pebble border, we define the clipped horizontal force $\overline{H}$:

1. If $B < \mu|F'_z|$ and $|\overline{F}'_y| \leq \sqrt{\mu^2(F')^2_z - B^2}$, then $\overline{H} = (\pm B, \overline{F}'_y, 0)$, where $\pm B$ takes its polarity from $\overline{F}'_x$.

2. If $B < \mu|F'_z|, \quad |\overline{F}'_y| > \sqrt{\mu^2(F')^2_z - B^2}$, and $B|\overline{F}'_y| \leq |\overline{F}'_x|\sqrt{\mu^2(F')^2_z - B^2}$, then $\overline{H} = (\pm B, \pm\sqrt{\mu^2(F')^2_z - B^2}, 0)$, where $\pm B$ takes the polarity from $\overline{F}'_x$, and $\pm\sqrt{\mu^2(F')^2_z - B^2}$ takes the polarity from $\overline{F}'_y$.

3. In the remaining case, $\overline{H} = \mu F'_z \dfrac{\overline{F}'_{xy}}{\|\overline{F}'_{xy}\|}$.

Complete Wheel Model

It seems that we now have collected everything needed to define a wheel model.

We have previous state S_t. If S_t is (2) or (3), we also have the following parameters:

- pebble position $\overline{p}_t$ and pebble speed $\overline{w}_t$ at time t (both in wheel coordinates).
- A contact surface defined by a normal $\overline{n}_t$ and its friction coefficient μ_t.

We are considering time $t + h$. We have to compute a force, and determine S_{t+h}. If $S_{t+h} \neq (1)$, we also have to determine $\overline{p}_{t+h}$ and $\overline{w}_{t+h}$.

State 1

We assume that (in wheel frame coordinates), the wheel runs from $(0,0,z_{min})$ to $(0,0,z_{max})$. Let $\overline{b}, q$ be its position and orientation. Let $\overline{x}_{min} = T_{\overline{b},q}(0,0,z_{min})$, $\quad \overline{x}_{max} = T_{\overline{b},q}(0,0,z_{max})$.

Get $(\lambda, \overline{n}) = \Phi(\overline{x}_{min}, \overline{x}_{max}, 0, 1, S)$ from the scenery solid S.

If $\lambda \geq 1$, we stay out of contact, so $S_{t+h} = (1)$, and force is zero.

Otherwise, set state $S_{t+h} = (3)$. (I think the state doesn't matter at this moment.) Set

$$\begin{cases} \overline{p}_{t+h} = \overline{x}_{min} + \lambda(\overline{x}_{max} - \overline{x}_{min}) \\ \overline{v}_{t+h} = \overline{0} \\ \overline{n}_{t+h} = \overline{n} \\ \overline{\mu}_{t+h} \text{ static friction coefficient of contact surface} \end{cases}$$

We assume that force stays zero. (It will be correct the next time.)

State 2,3

Let $(\overline{b}, q)$ be the current position and orientation of the wheel frame, let $\overline{v}$ be its current speed.

Convert $\overline{p}_t$ and $\overline{v}_t$ into wheel frame coordinates:

$$\begin{cases} \overline{p} = T^{-1}_{\overline{b},q}(\overline{p}_t) \\ \overline{w} = f_{q^{-1}}(\overline{w}_t - \overline{v}) \end{cases}$$

Let $\overline{F} = \overline{F}(\overline{p}, \overline{w})$, let $\overline{F}_{cl} = \overline{F}_{cl}(\overline{F}, \mu_t, \overline{n})$.

If $\overline{F}_{cl} = \overline{F}$, then the next state will be State 3 (standing), so that $S_{t+h} = (3), \quad \overline{p}_{t+h} = \overline{p}_t, \quad \overline{w}_{t+h} = \overline{0}, \quad \overline{n}_{t+h} = \overline{n}_t, \quad \mu_{t+h} = \mu_s.$
Wheel force is equal to $f_q(\overline{F})$. (Force from pebble, transformed into world coordinates.)

State 2,3, Slipping

If $\overline{F}_{cl} \neq \overline{F}$, then next state will be State 2 (slipping). Convert $\overline{p}_t$ and $\overline{v}_t$ into wheel frame coordinates:

$$\begin{cases} \overline{p} = T^{-1}_{\overline{b},q}(\overline{p}_t) \\ \overline{w} = f_{q^{-1}}(\overline{w}_t - \overline{v}) \end{cases}$$

We would like to move the pebble to a position $\overline{p}'$ where the force equals exactly $\overline{F}_{cl}$. This is not easy, because distance from surface (and with it the normal force) may be different at the new point, we may be in contact with another surface, and we would have to take the speed $\overline{w}'$ into account. Solving the system of equations seems unrealistic. Iterating to the right point is possible, but expensive.

State 2,3, Slipping

In slipping state, we expect the pebble to move more or less at the same time as the wheel frame, so that we probably can neglect the speed. This gives:

$$(x, y, 0) = \frac{\overline{F}_{cl,xy}}{k_{xy}},$$

where $\overline{F}_{cl,xy}$ is the horizontal component of $\overline{F}_{cl}$.

Let $\overline{x}_{min} = T_{\overline{b},q}(x, y, z_{min})$, $\overline{x}_{max} = T_{\overline{b},q}(x, y, z_{max})$. Let

$$(\lambda, \overline{n}) = \Phi(\overline{x}_{min}, \overline{x}_{max}, 0, 1, S).$$

If $\lambda \geq 1$, we fell over the border or managed to take off, and $S_{t+h} = (1)$.

State 2,3, Slipping

$$\begin{cases} S_{t+h} = (2) \\ \overline{p}_{t+h} = \overline{x}_{min} + \lambda(\overline{x}_{max} - \overline{x}_{min}) \\ \overline{w}_{t+h} = \dfrac{\overline{p}_{t+h} - \overline{p}_t}{h} \\ \overline{n}_{t+h} = \overline{n} \\ \overline{\mu}_{t+h} = \text{dynamic friction coefficient of contact surface} \end{cases}$$

Finally, the force equals $f_q(\overline{F}_{cl})$. (The clipped force transformed into world coordinates.)

Disclaimer: All of this has to be implemented, in order to see if the results are realistic. Don't trust any theory that was not implemented and tested!