June 4th 2013

Collision Detection and Response Among Rigid Bodies

Simone Chesi
Veronica Pellegrini

U.S. Naval Postgraduate School, Monterey, California

NAVAL POSTGRADUATE SCHOOL

Spacecraft Robotics LABORATORY

Outlines

- **Introduction**
- **Collision Detection**
- **Collision Response**
- **Simulink Collision Block**

Introduction

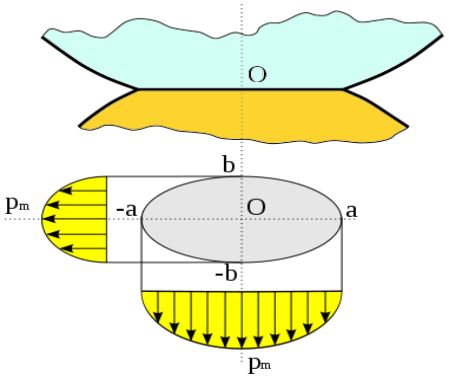
CONTACT MECHANICS: *Study of the deformation of solids that touch each other in one or more points.*

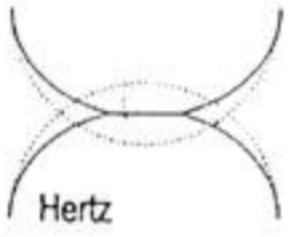
- H. Hertz, Über die berührung fester elastischer Körper "On the contact of rigid elastic solids". In: Miscellaneous Papers. Jones and Schott, Editors, J. reine und angewandte Mathematik 92, Macmillan, London (1882).

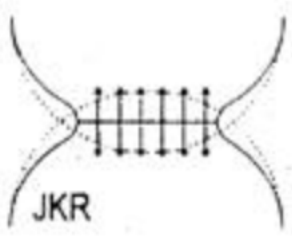
- Hertzian theory provides contact stress as a function of the normal contact force, the radii of curvature of both bodies and the modulus of elasticity of both bodies.

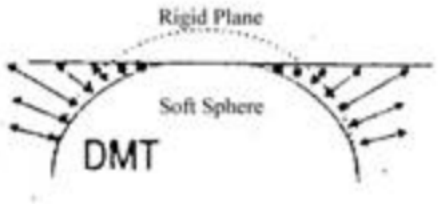
- Other models: *1970, JKR* (Johnson, Kendall, Roberts) Theory, (Adhesive contact). DMT theory consider effects outside elastic contact regime.

- Other models of contact mechanics have been introduced and all these models deal with nonlinear equation and/or special cases.









Introduction: Physics Engines, Forces or Impulses?

There are two kind of physics engines:

- Real Time Simplified Models
- High Precision Complex Models mostly nonlinear (Long time to compute)

Collision Detection

- Collisions occurs when two objects in free flight impact each other and bounce apart. In rigid body simulations, collisions are usually handled by applying *impulses* that *discontinuously modify the velocities of the bodies at the time of collision*.

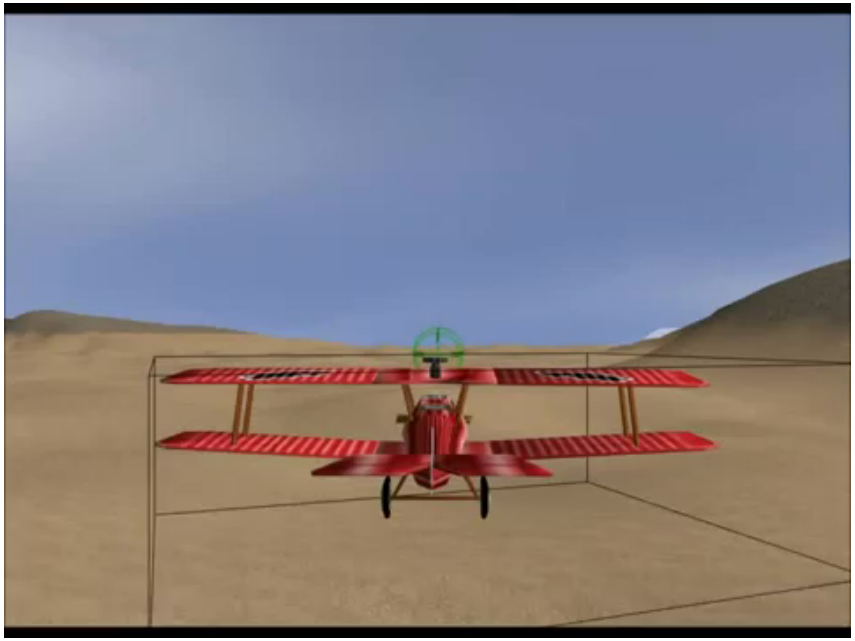
- A rigid body collision should not to be confused with *collision detection*, the latter is a problem of determining if and how two objects are intersecting;

- Directly testing the geometry of two objects for collision against each other can be computationally expensive.

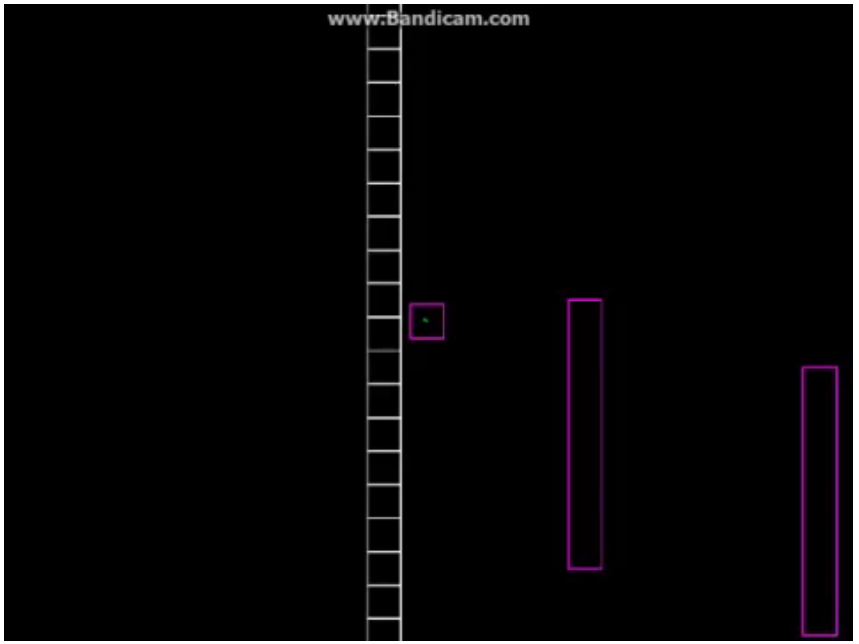
- A *bounding volume* (BV) is a single volume encapsulating one or more objects of more complex nature. Desirable properties are:
  - ✓ Inexpensive intersection tests;
  - ✓ Tight fitting;
  - ✓ Inexpensive to compute;
  - ✓ Easy to rotate and transform.

Collision Detection

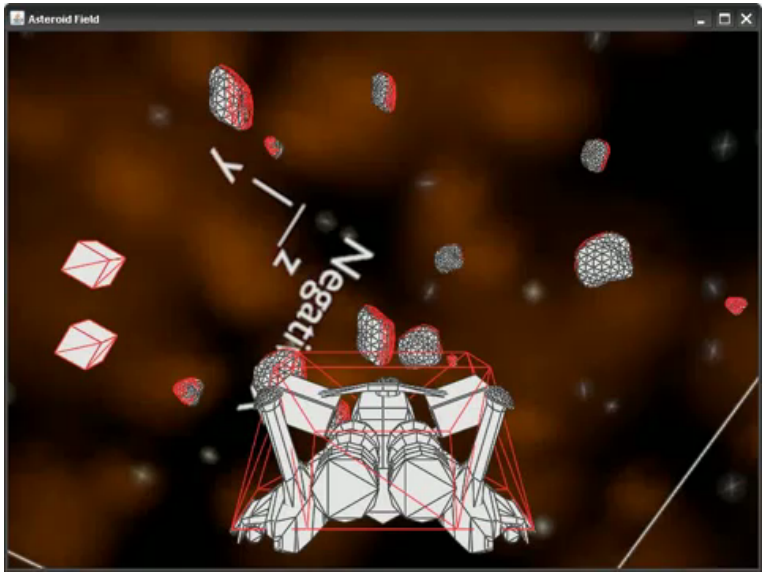
AABB



OBB



Convex Hull



Collision Detection: Closest Points

- Closest-point queries are some of the most powerful of collision queries;

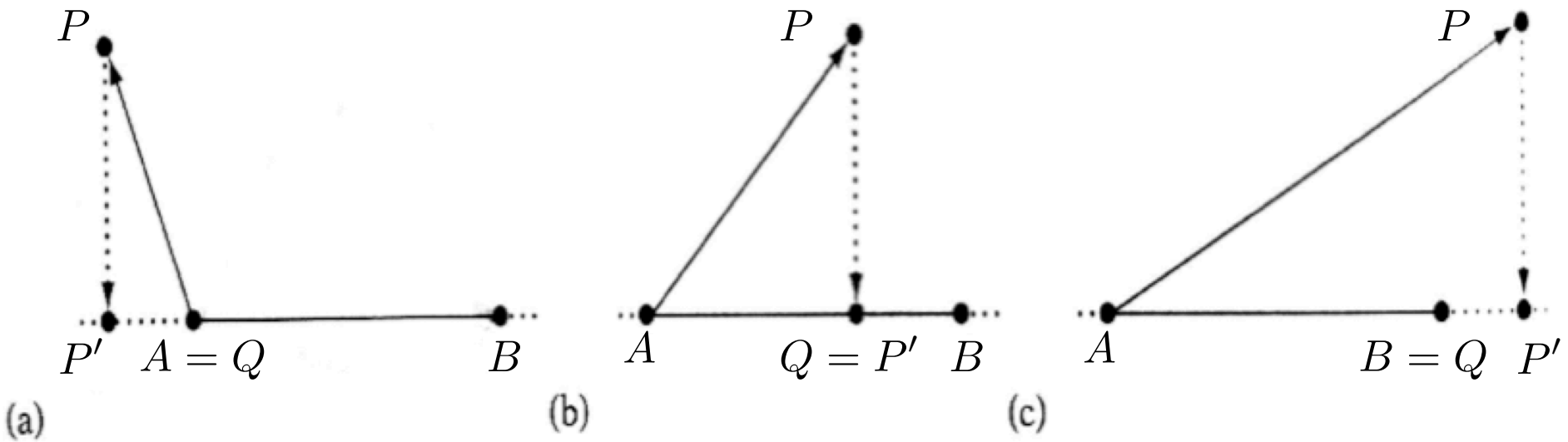
- Given the closest points between two objects, the distance between the objects is obtained ⟶ collision can be determined!

- Obtaining the closest points can be seen as: a minimization problem; a *geometrical one*.

- Following we will illustrate how the closest points can be obtained for various geometric objects (2D):

  - ✓ Point to line;
  - ✓ Line to line;
  - ✓ General polygons.

Collision Detection: Point to Line Segment



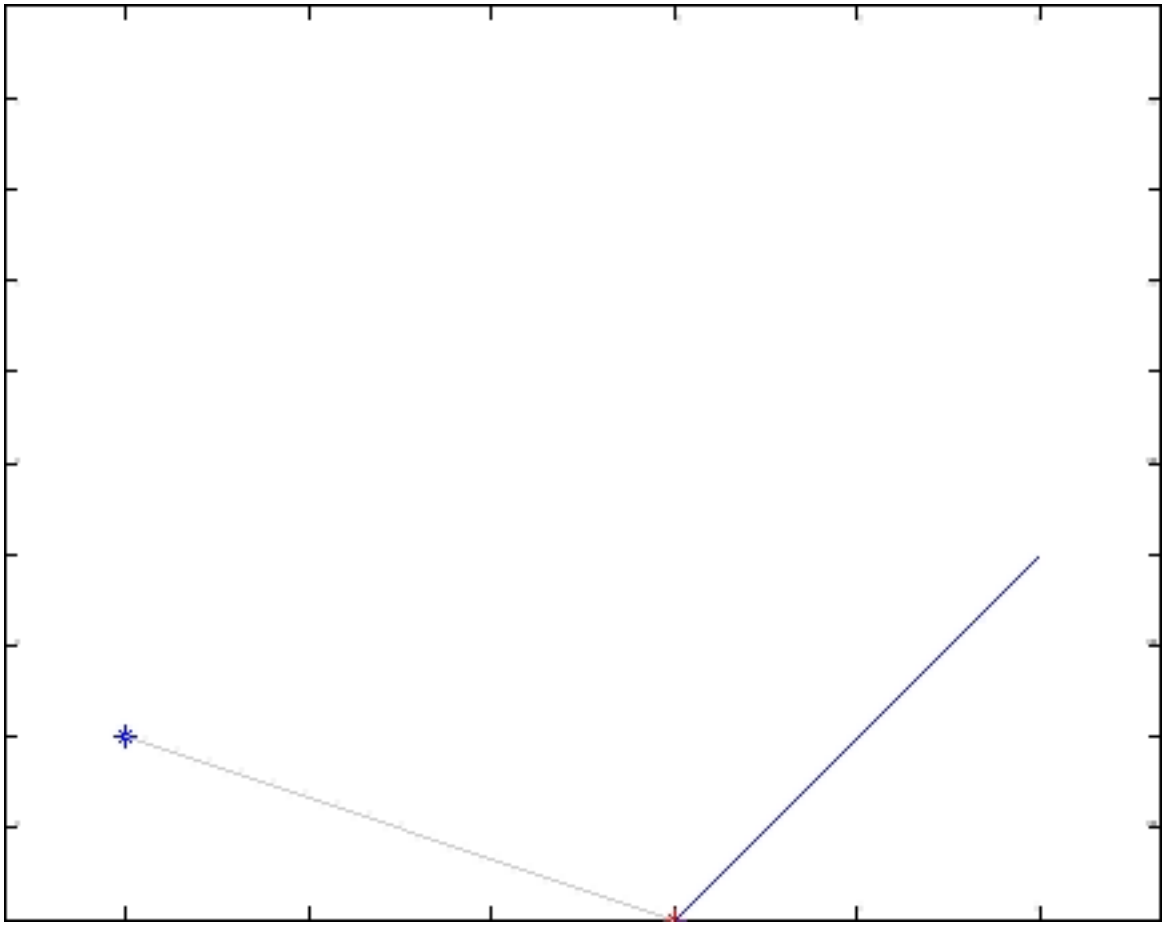
$$Q = A + t_d(B - A) \quad \text{where} \quad t_d \in [0, 1]$$

$$t_d = \frac{(P - A) \cdot \boldsymbol{s}}{||B - A||} \quad \text{where} \quad \boldsymbol{s} = \frac{(B - A)}{||B - A||}$$

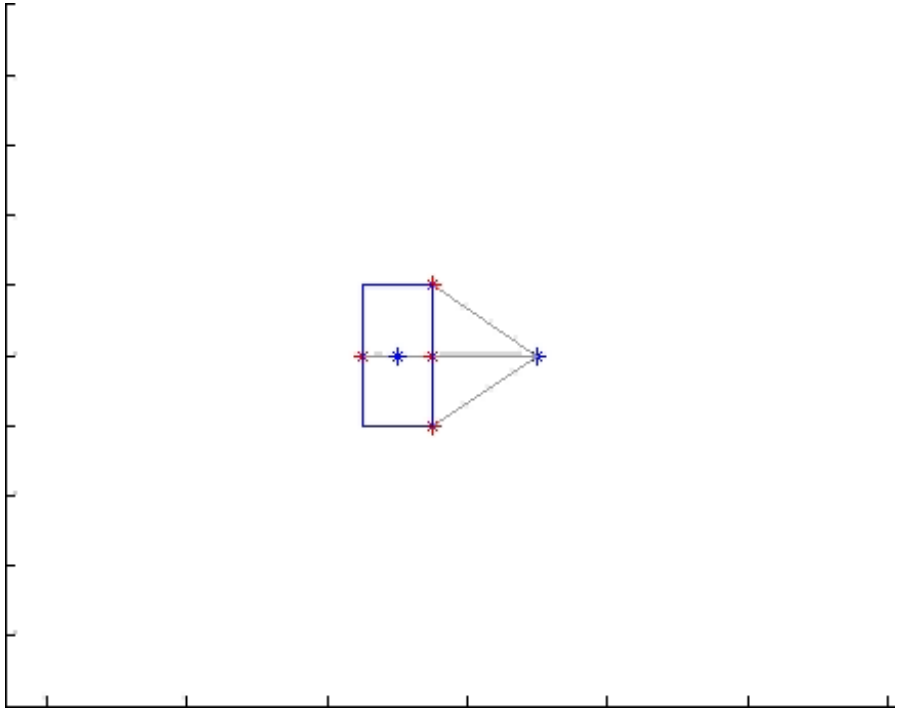
Unit vector in the direction of AB

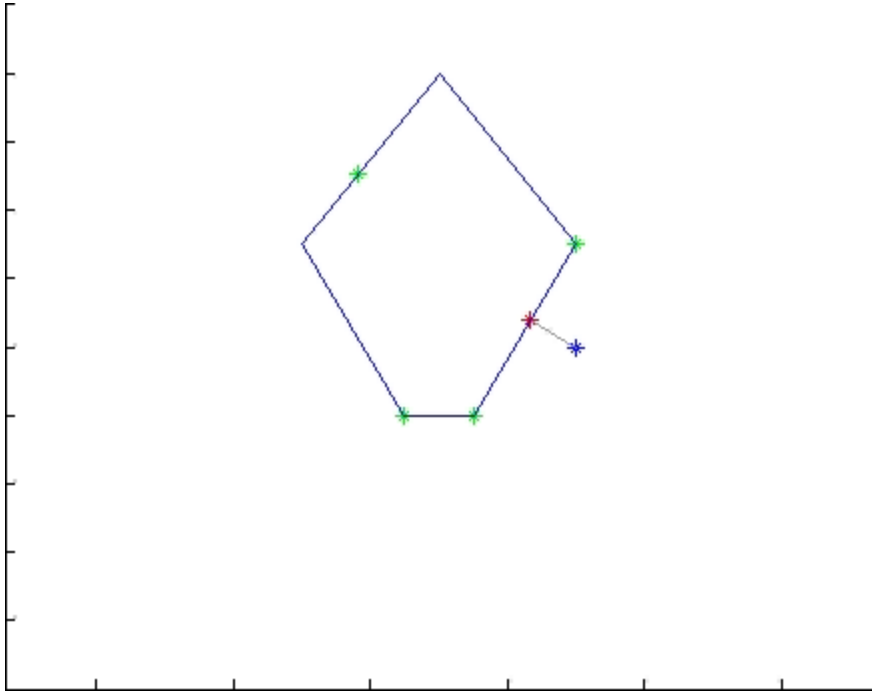
- Projection of P on AB:
  - ✓ P' is outside AB on its left (a): Q = A → $t_d = 0$
  - ✓ P' is inside AB (b): Q = P' → $t_d \in [0; 1]$
  - ✓ P' is outside AB on its right (c): Q = B → $t_d = 1$

Collision Detection: Point to Line



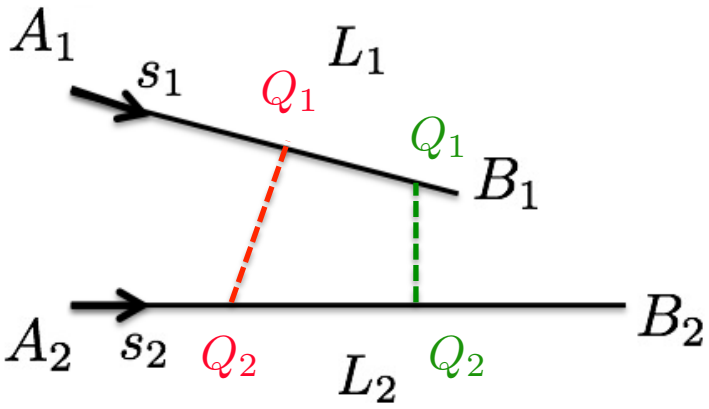
Collision Detection: Point to Line





Collision Detection: Two Line Segments

- Take a general point Q2 on L2, we want closest point Q1 to Q2 on L1:



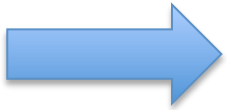
$$Q_1 = A_1 + t_{d1}(B1 - A1)$$

$$t_{d1} = \frac{(Q_2 - A_1) \cdot \boldsymbol{s_1}}{||B_1 - A_1||}$$

- Take a general point Q_1 on L_1, we want closest point Q2 to Q1 on L_2:

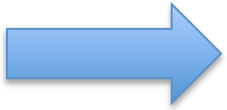
$$Q_2 = A_2 + t_{d2}(B_2 - A_2)$$

$$t_{d2} = \frac{(Q_1 - A_2) \cdot \boldsymbol{s_2}}{||B_2 - A_2||}$$



We want:
closest point on L_2 to L_1 = closest point on L_1 to L_2

Collision Detection: Two Line Segments



We want:
closest point on L2 to L1 = closest point on L1 to L2

- therefore, since:

$$Q_1 = A_1 + t_{d1}(B1 - A1) \qquad Q_2 = A_2 + t_{d2}(B_2 - A_2)$$

- we have:

$$t_{d1} = \frac{(Q_2 - A_1) \cdot \boldsymbol{s}_1}{||B_1 - A_1||} = \frac{[A_2 + t_{d2}(B_2 - A_2) - A_1] \cdot \boldsymbol{s}_1}{||B_1 - A_1||}$$
$$t_{d1} = \frac{(Q_1 - A_2) \cdot \boldsymbol{s}_2}{||B_2 - A_2||} = \frac{[A_1 + t_{d1}(B_1 - A_1) - A_2] \cdot \boldsymbol{s}_2}{||B_2 - A_2||}$$

- By algebraic manipulation:

$$t_{d1} = \frac{bt_{d2} - c}{a}$$
$$t_{d2} = \frac{bt_{d1} + f}{e}$$

$$a = (B_1 - A_1) \cdot (B_1 - A_1)$$
$$b = (B_1 - A_1) \cdot (B_2 - A_2)$$
$$c = (B_1 - A_1) \cdot (A_1 - A_2)$$
$$e = (B_2 - A_2) \cdot (B_2 - A_2)$$
$$f = (B_2 - A_2) \cdot (A_1 - A_2)$$

Collision Detection: Two Line Segments

- Solving for the two unknowns:

$$a = (B_1 - A_1) \cdot (B_1 - A_1)$$
$$b = (B_1 - A_1) \cdot (B_2 - A_2)$$
$$c = (B_1 - A_1) \cdot (A_1 - A_2)$$
$$e = (B_2 - A_2) \cdot (B_2 - A_2)$$
$$f = (B_2 - A_2) \cdot (A_1 - A_2)$$

$$t_{d1} = \frac{(bf - ce)}{ae - b^2}$$

$$t_{d2} = \frac{f - bc}{ea - b^2}$$

- Once td1 is obtained, we evaluate td2 and consider that:

$$if \quad t_{d2} < 0 \quad \rightarrow \quad t_{d2} = 0 \quad \rightarrow \quad t_{d1} = \frac{bt_{d2} - c}{a}$$

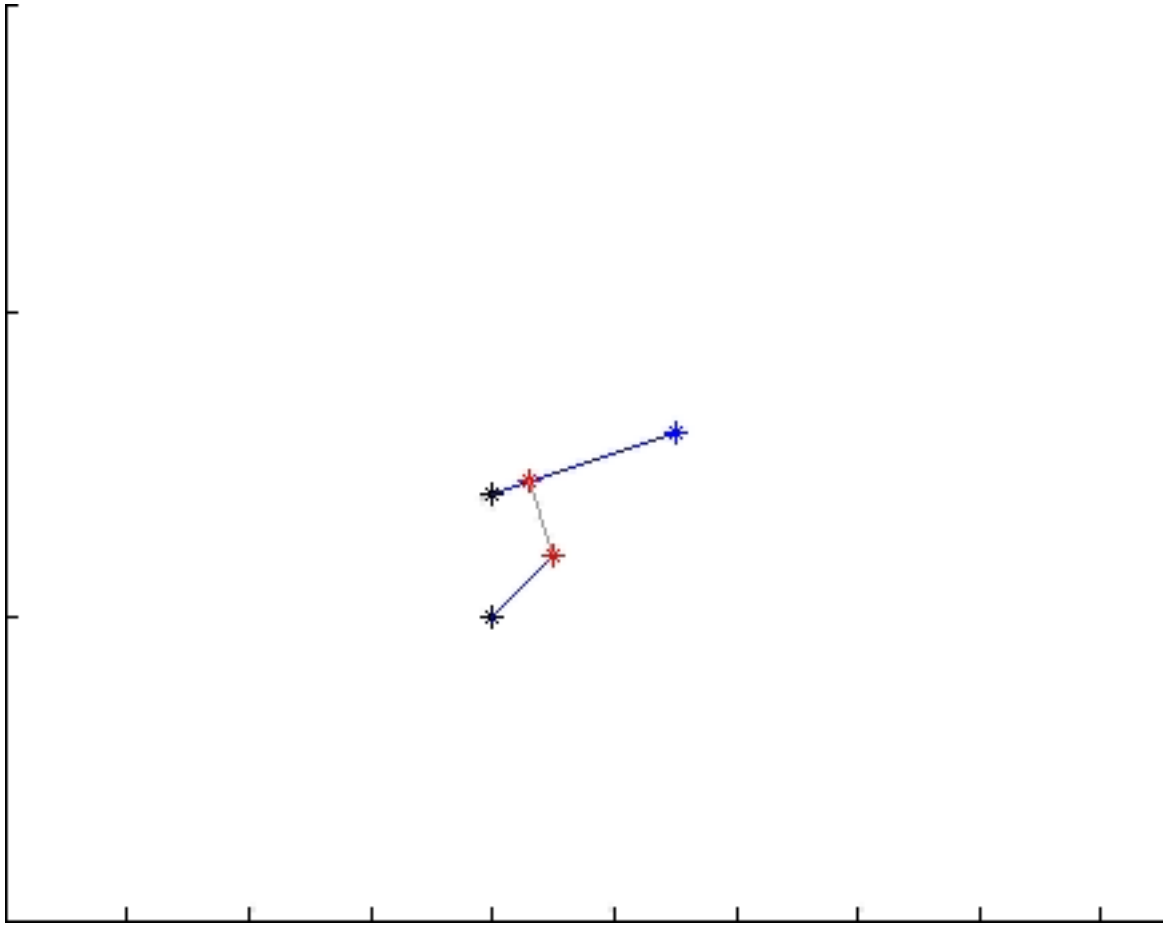
$$if \quad t_{d2} > 1 \quad \rightarrow \quad t_{d2} = 1 \quad \rightarrow \quad t_{d1} = \frac{bt_{d2} - c}{a}$$

- Also, we need to consider that:

$$if \quad (ea - b^2) = 0 \quad \rightarrow$$ The two lines are *parallel*

In this case we fix the value of one parameter and evaluate the other.

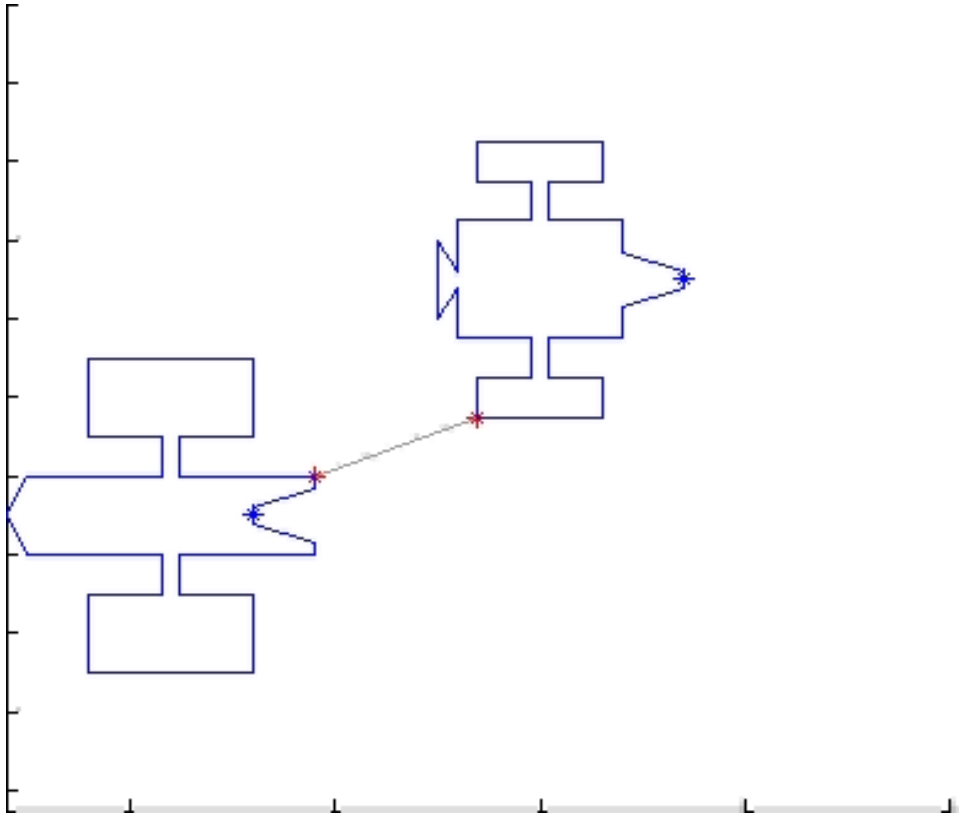
Collision Detection: Two Line Segments



Collision Detection: Two Polygons

- By unify all concepts we describes:

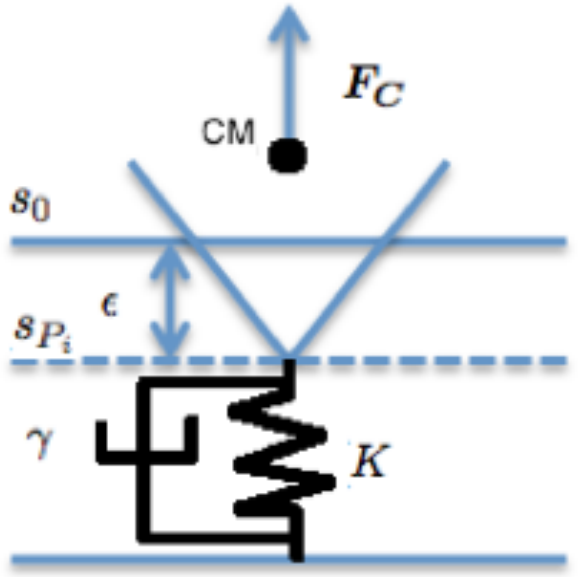
1. Evaluate the distance of each line on one polygons to each line on the second polygon;
2. Consider the minimum distance: this will be the “most dangerous distance” to monitor for collision purpose.



Collision Response: Mass-Spring Model

Collision Response: OBB

The body is approximated as a bounding box.



The collision force is:

$$\boldsymbol{F}_c = -[K(s_{p_i} - s_0) + \gamma \dot{s}_{p_i}]\boldsymbol{n}$$

If the body collides with the surface the dynamics becomes:

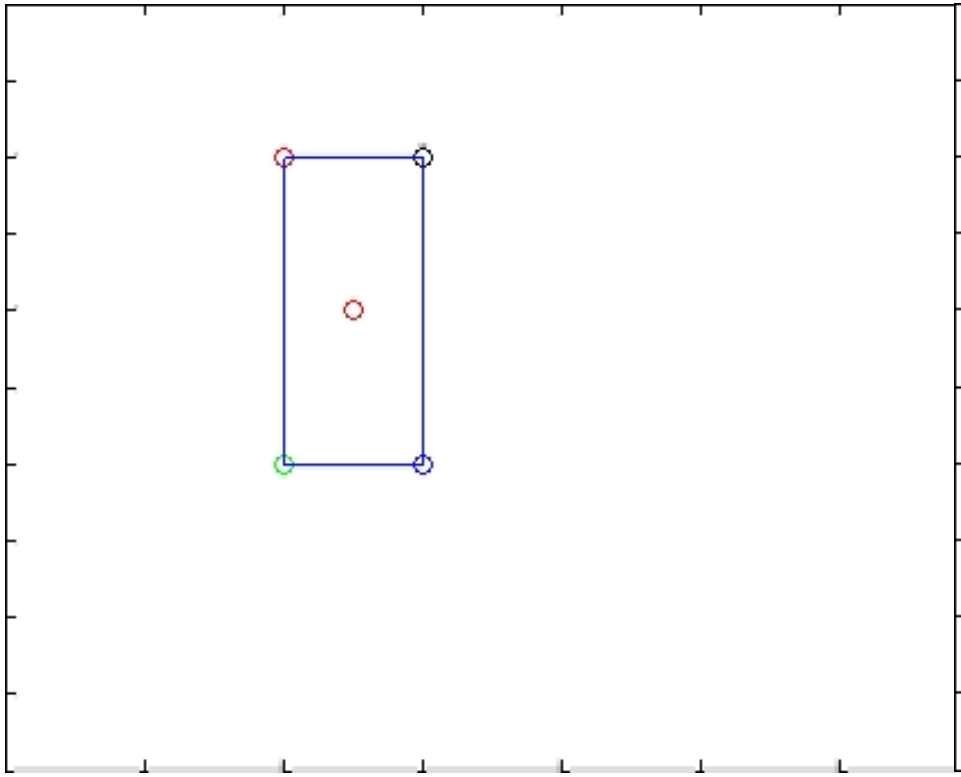
$$\ddot{\boldsymbol{X}} = \frac{\boldsymbol{F}_c}{m}$$

K = Spring Constant

γ = Damping Coefficient

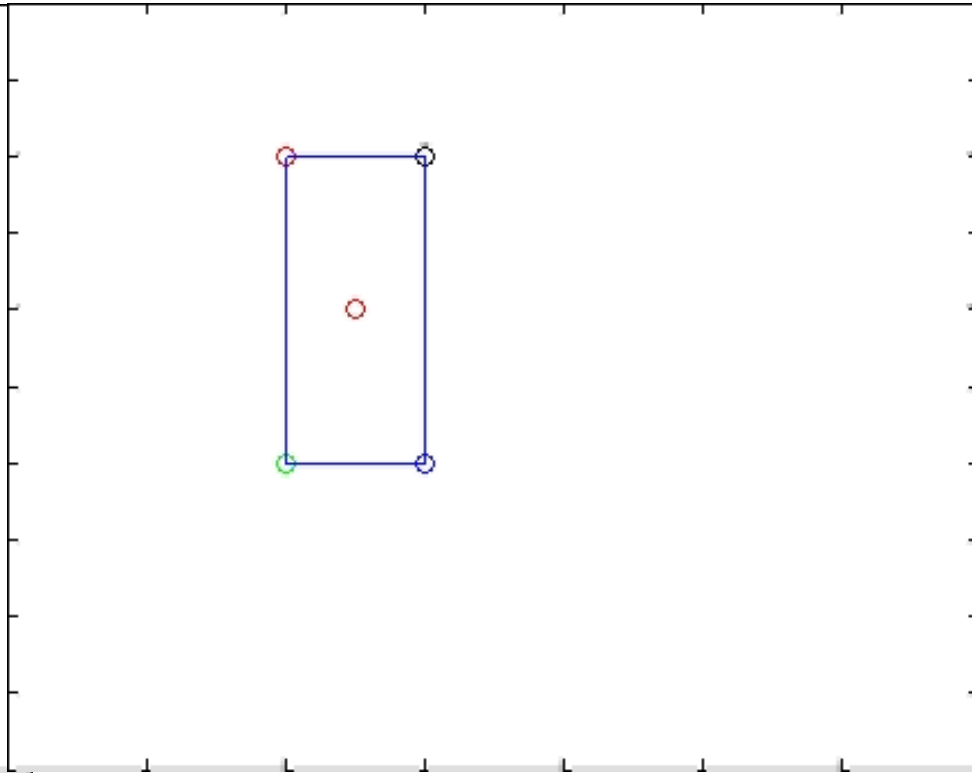
Collision Response: Mass-Spring Model, WHY NOT?

Low value of spring constant



No rigid body collision

High value of spring constant



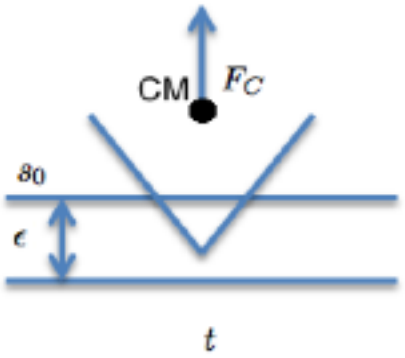
No elastic collision

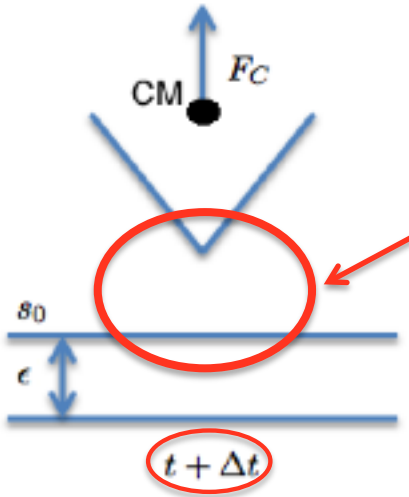
Collision Response: Mass-Spring Model, WHY NOT?

In order to reduce penetration → Large value of *k*

In order to reduce the effect of large *k* → Large interpenetration

Small time step integration needed!! → Difficult during Real-Time Simulation





The body is out and force is still applied!! ENERGY IS NOT CONSERVED!!

Collision Response: Impulsive Method

- In rigid body simulations, collisions are usually handled by applying impulses that discontinuously modify the velocities of the bodies at the time of collision;

- This change in body's momentum, called *impulse* is:

$$\boldsymbol{j} = \int_{t_0}^{t_f} \boldsymbol{f}\, dt$$

Collisions between real objects are not instantaneous ➔ impractical to simulate the compression and expansion!!

Therefore, by assuming that a suitable f can be found such that the limit exists and is:

$$\boldsymbol{j} = \lim_{t_0 \to t_f} \int_{t_0}^{t_f} \boldsymbol{f}\, dt$$

This produced the idea of *instantaneous impulse.*

- Therefore: *Impulse*, is an *instantaneous* transfer of momentum between two colliding rigid bodies, with consequent *instantaneous* change in velocity.

Collision Response: Impulsive Method

In a close system not subjected to external forces or torques the kinetic energy, angular momentum and linear momentum are conserved

$$\frac{d}{dt}\boldsymbol{P} = 0$$

$$\frac{d}{dt}\boldsymbol{\Gamma} = 0$$

$$\frac{d}{dt}E = 0$$

For a system composed by *n* rigid bodies, the total energy, angular and linear momentum are

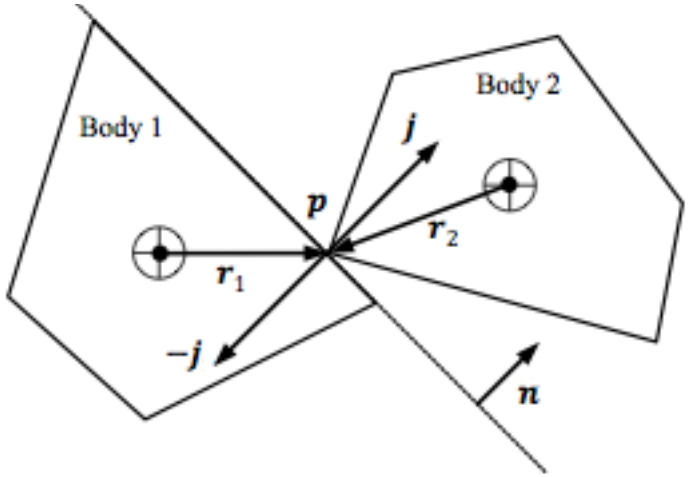
$$P = \sum_{i=1}^{n} \boldsymbol{p}_i = \sum_{i=1}^{n} m_i \boldsymbol{v}_i$$

$$\boldsymbol{\Gamma} = \sum_{i=1}^{n} \boldsymbol{r}_{cm_i} \times (m_i \boldsymbol{v}_i) + \boldsymbol{I}^I \boldsymbol{\omega}_i$$

$$E = \sum_{i=1}^{n} \frac{1}{2} m_1 \boldsymbol{V}_i^T \boldsymbol{V}_i + \frac{1}{2} \boldsymbol{\omega}_i^T \boldsymbol{I} \boldsymbol{\omega}$$

Collision Response: Impulsive Method

Important: *The total linear and angular momentum are conserved in case of* ***elastic*** *or* ***inelastic*** *collision* ***(e.g. docking of two spacecraft)****. The kinetic energy is conserved only in case of elastic collision.*



Consider now a generic collision between two bodies. The change in linear momentum is equal to the impulse on each body:

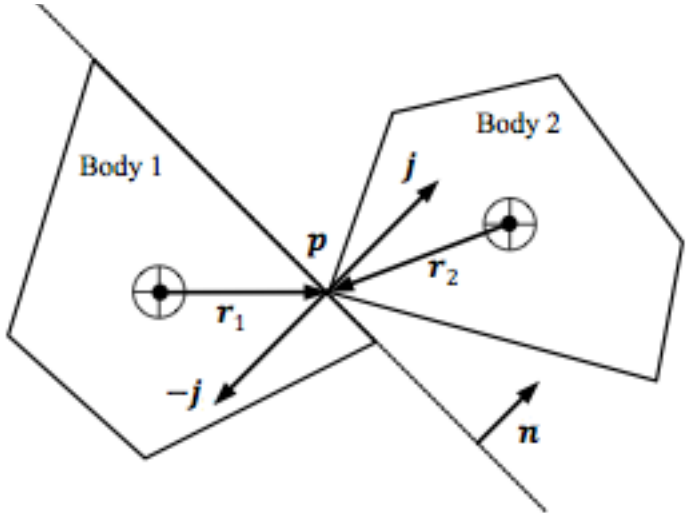
$$m_1(\boldsymbol{v}_1' - \boldsymbol{v}_1) = -\boldsymbol{j}$$
$$m_1(\boldsymbol{v}_2' - \boldsymbol{v}_2) = \boldsymbol{j}$$

$$\boldsymbol{v}_1' = \boldsymbol{v}_1 - \frac{j}{m_1}\boldsymbol{n}$$
$$\boldsymbol{v}_2' = \boldsymbol{v}_2 + \frac{j}{m_2}\boldsymbol{n}$$

Where:

$$\boldsymbol{j} = j\boldsymbol{n} \qquad j \in R \qquad \boldsymbol{n} \in R^3$$

Collision Response: Impulsive Method



Similarly, the change in angular momentum is equal to the impulse moment. Therefore, the angular velocity after the collision are

$$\boldsymbol{\omega}_1' = \boldsymbol{\omega}_1 - \frac{\boldsymbol{r}_1 \times j\boldsymbol{n}}{I_1}$$

$$\boldsymbol{\omega}_2' = \boldsymbol{\omega}_2 + \frac{\boldsymbol{r}_2 \times j\boldsymbol{n}}{I_2}$$

The point $\boldsymbol{p}$ at the instant of collision is common to both bodies, denoted by $\boldsymbol{p}1$, $\boldsymbol{p}2$ such that $\boldsymbol{p}1 = \boldsymbol{p}2 = \boldsymbol{p}$. The respective velocities is:

$$\boldsymbol{v}_{P_i} = \boldsymbol{v}_i + \boldsymbol{\omega}_i \times \boldsymbol{r}_i \qquad i = 1, 2$$

Collision Response: Impulsive Method

The coefficient of restitution *e* relates the pre-collision relative velocity to the post-collision relative velocity along the contact normal as:

$$\boldsymbol{v}'_r \cdot \boldsymbol{n} = -e\boldsymbol{v}_r \cdot \boldsymbol{n} \qquad 0 \leq e \leq 1$$

where:

$$\boldsymbol{v}_r = \boldsymbol{v}_{P_1} - \boldsymbol{v}_{P_2}$$
$$\boldsymbol{v}'_r = \boldsymbol{v}'_{P_1} - \boldsymbol{v}'_{P_2}$$

Substituting into this equation all the expressions we found before and solving for the reaction impulse magnitude *j* yields (after algebraic manipulations):

$$j = \frac{-(1+e)\boldsymbol{v}_r \cdot \boldsymbol{n}}{\frac{1}{m_1} + \frac{1}{m_2} + (\frac{1}{I_1}(\boldsymbol{r}_1 \times \boldsymbol{n}) \times \boldsymbol{r}_1 + \frac{1}{I_2}(\boldsymbol{r}_2 \times \boldsymbol{n}) \times \boldsymbol{r}_2) \cdot \boldsymbol{n}}$$

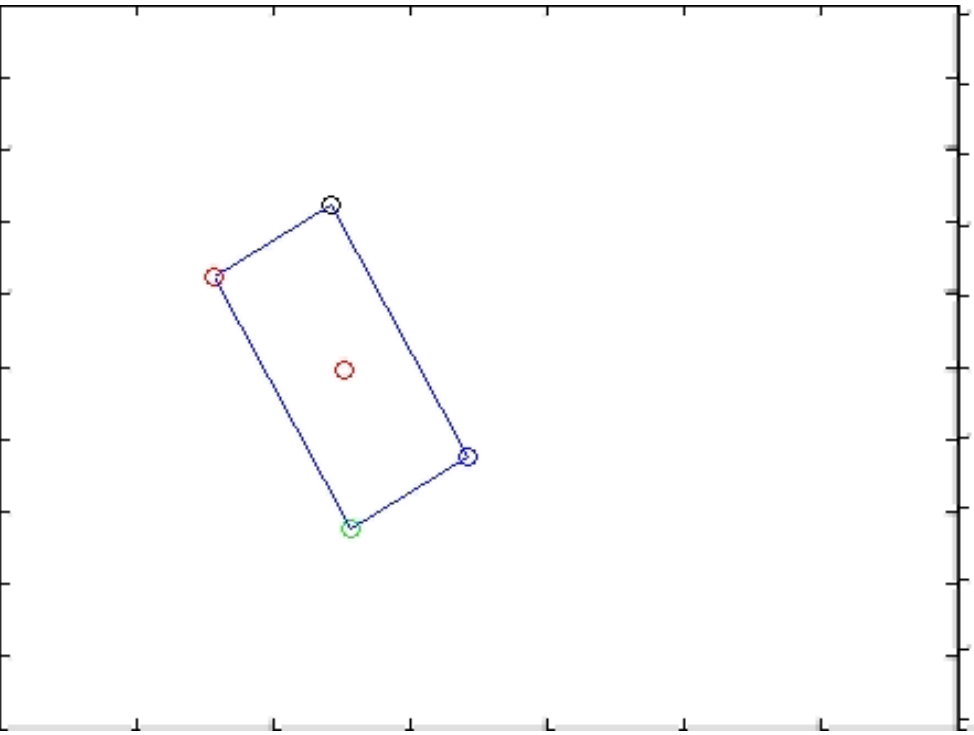
If one of the two body is massive w.r.t. the other, i.e. $I_1 \to \infty \;\; m_1 \to \infty$

$$j = \frac{-(1+e)\boldsymbol{v}_r \cdot \boldsymbol{n}}{\frac{1}{m_2} + (\frac{1}{I_2}(\boldsymbol{r}_2 \times \boldsymbol{n}) \times \boldsymbol{r}_2) \cdot \boldsymbol{n}}$$

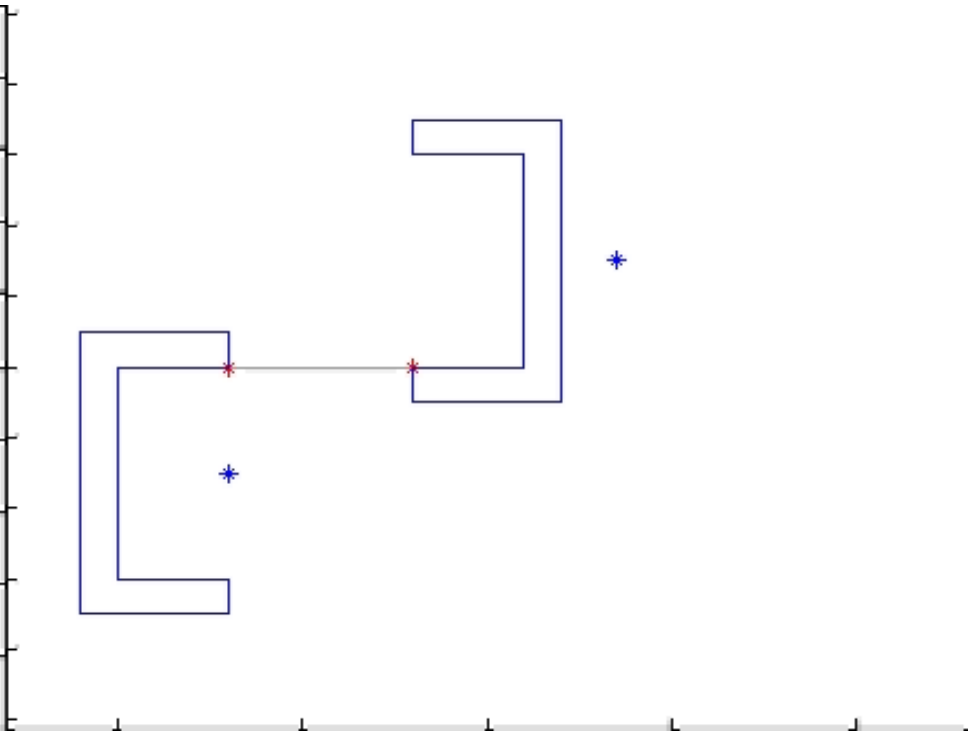
Using the equations above it is possible to prove analytically that the linear and angular momentum are conserved during collisions.

Testing Conservation of Energy and Momentum

Conservation of Energy



Conservation of Momentum



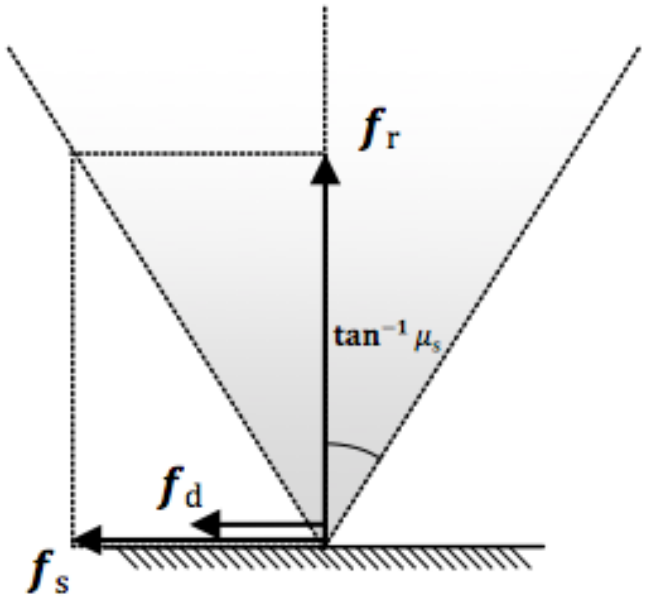
Collision Response: Impulsive Method

IMPULSE-BASED FRICTION (Coulomb Friction Model):

- $\mu_s, \mu_d \in R$ are the static and dynamic friction coefficients such that $\mu_s \geq \mu_d$.
- The coefficients of friction is an ***empirical measurement .***

- Therefore the static and dynamic friction force magnitudes f_s, f_d are computed in terms of the reaction force magnitude f_r.

- $f_e, f_f \in R^3$ are the external forces and the friction force acting on the rigid body



$$f_s = \mu_s ||\boldsymbol{f}_r||$$
$$f_d = \mu_d ||\boldsymbol{f}_r||$$

and:

$$\boldsymbol{f}_f = \begin{cases} -(\boldsymbol{f}_e \cdot \boldsymbol{t})\boldsymbol{t} & \boldsymbol{v}_r = \boldsymbol{0} \quad \boldsymbol{f}_e \cdot \boldsymbol{t} \leq f_s \\ -f_s \boldsymbol{t} & \boldsymbol{v}_r = \boldsymbol{0} \quad \boldsymbol{f}_e \cdot \boldsymbol{t} > f_s \\ -f_d \boldsymbol{t} & \boldsymbol{v}_r \neq \boldsymbol{0} \end{cases}$$

F: sum of all external forces on the body

Collision Response: Impulsive Method

- By adapting the argument for instantaneous impulses, an impulse-based version of the Coulomb friction model may be derived:

$$j_s = \mu_s j$$
$$j_d = \mu_d j$$

where:

$$\boldsymbol{j} = j\boldsymbol{n}$$

- Therefore similarly to force (by integration) we have:

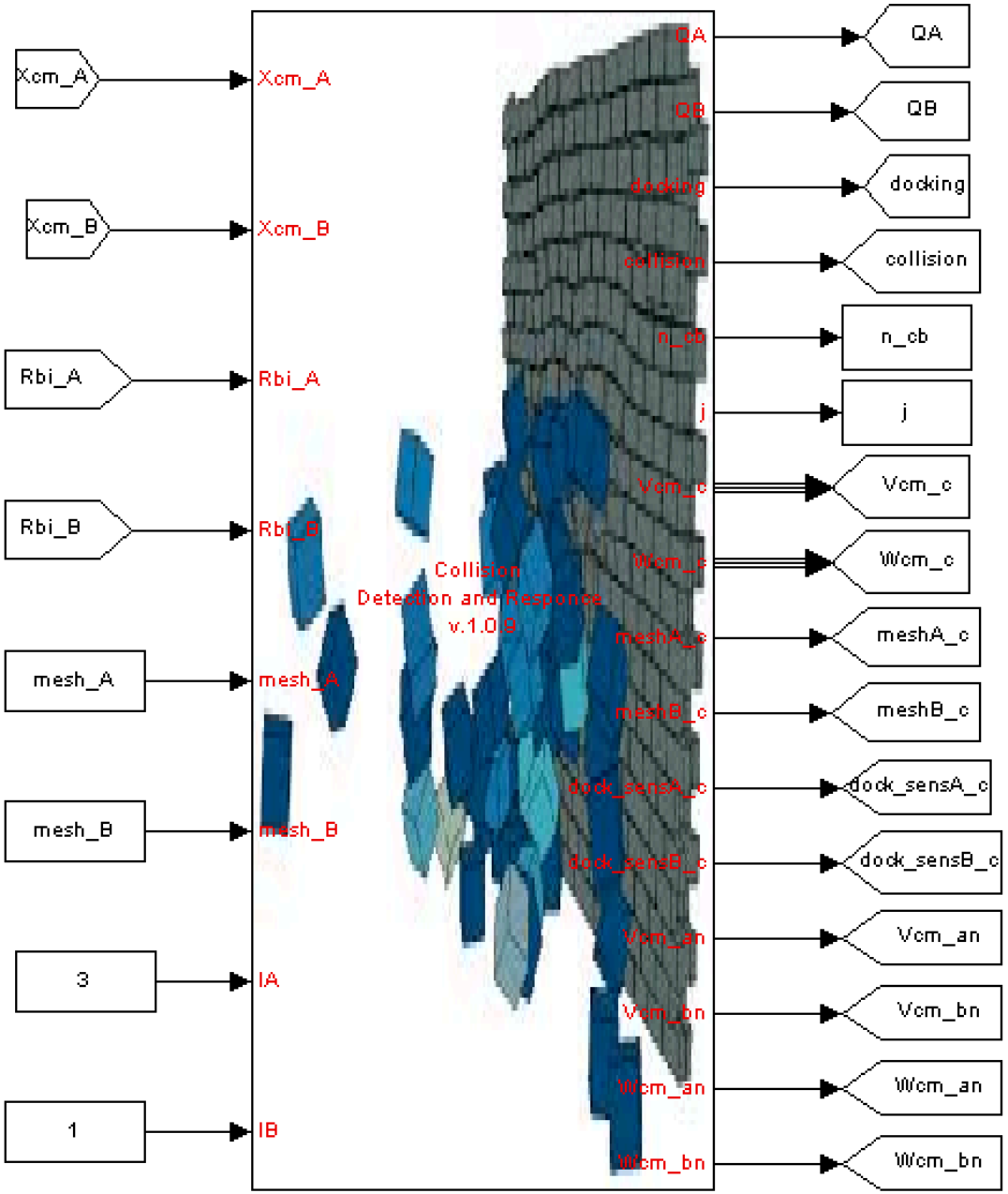
$$\boldsymbol{j}_f = \begin{cases} -(m\boldsymbol{v}_r \cdot \boldsymbol{t})\boldsymbol{t} & \text{if } \boldsymbol{v}_r = 0, \quad m\boldsymbol{v}_r \cdot \boldsymbol{t} \leq j_s \\ -j_s\boldsymbol{t} & \text{if } \boldsymbol{v}_r = 0, \quad m\boldsymbol{v}_r \cdot \boldsymbol{t} > j_s \\ -j_d\boldsymbol{t} & \text{if } \boldsymbol{v}_r \neq 0 \end{cases}$$

- For simplicity, we will consider:

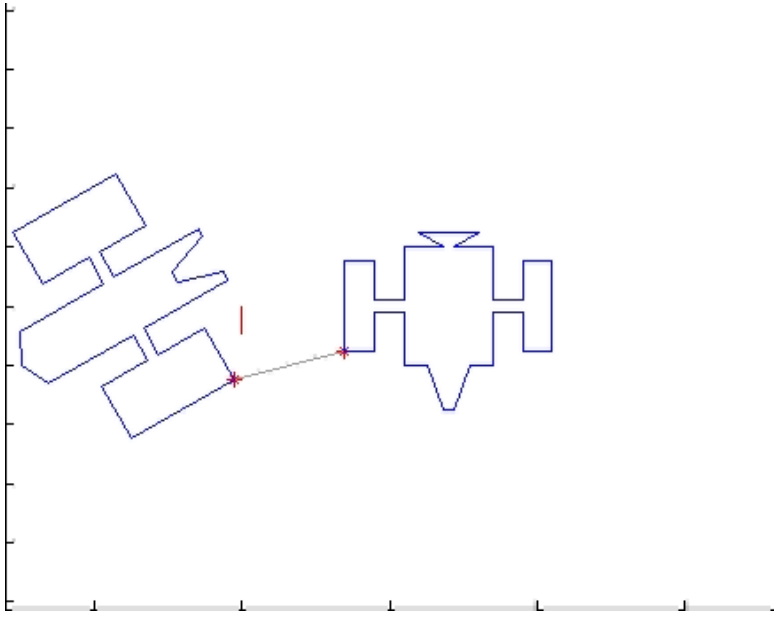
$$\boldsymbol{j}_f = 0$$

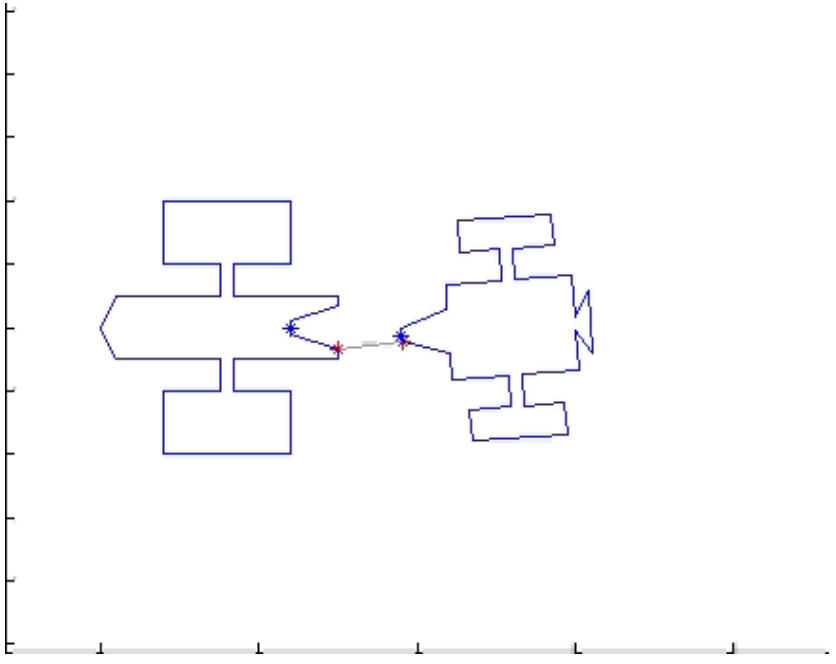
therefore the impulse applied is purely normal to the body surface at the point of contact.

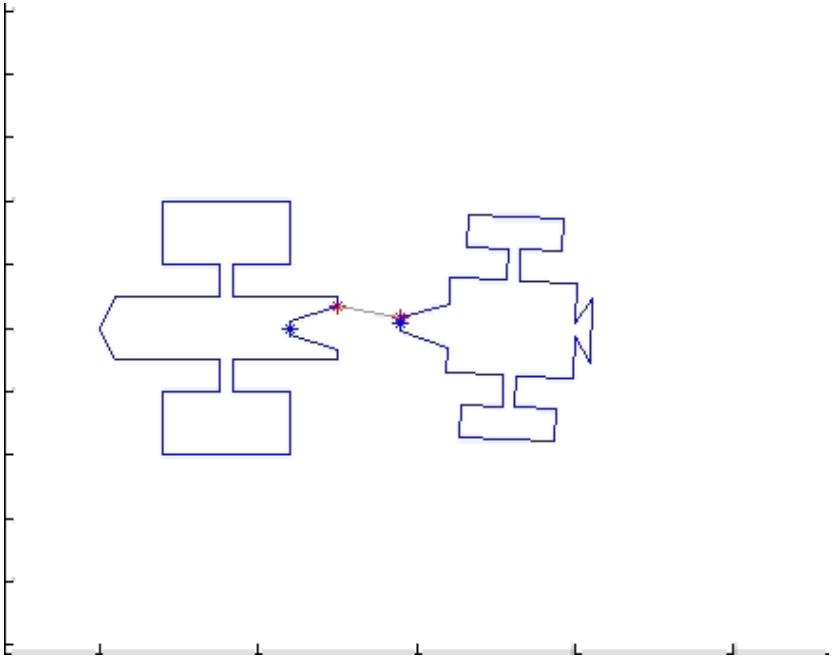
Simulink Collision Block



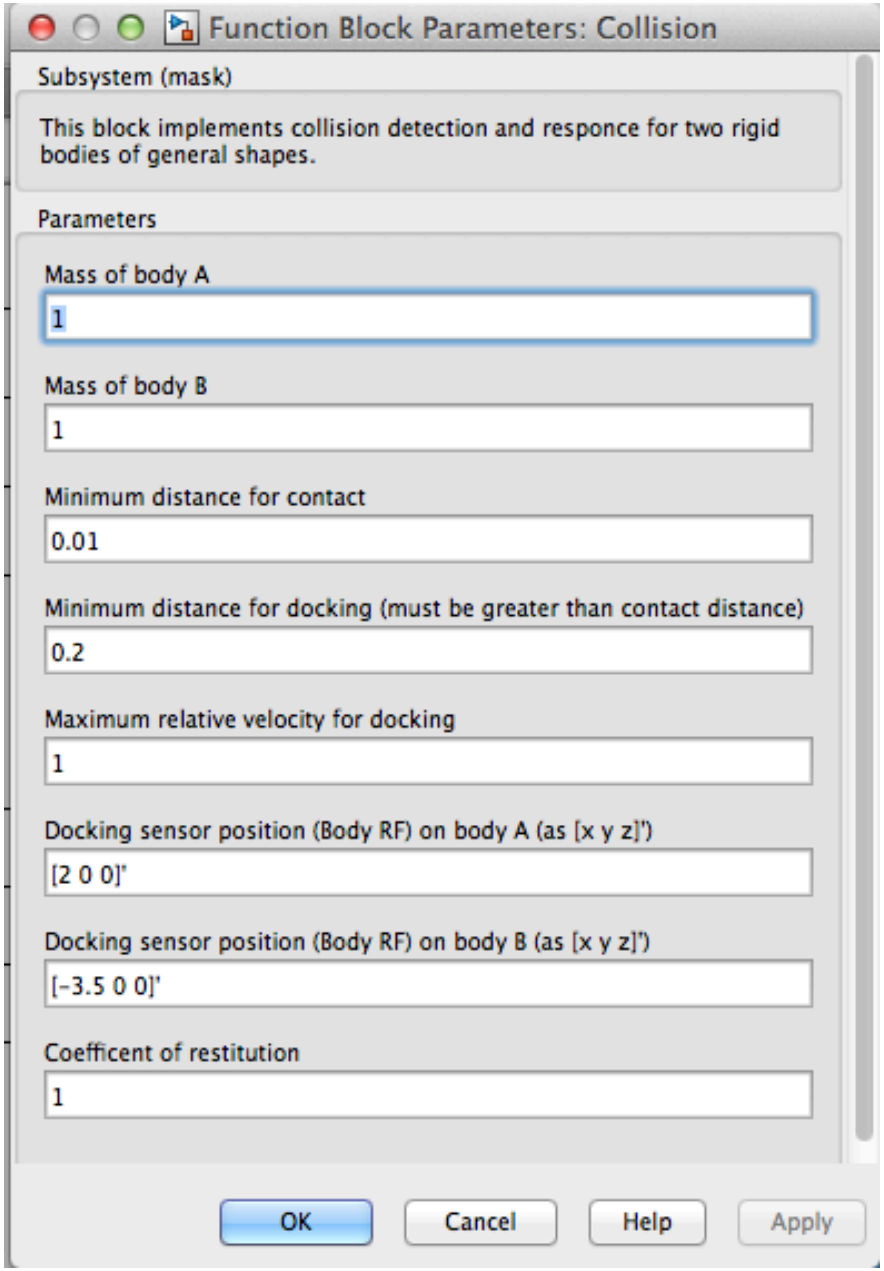
Simulink Collision Block: Examples

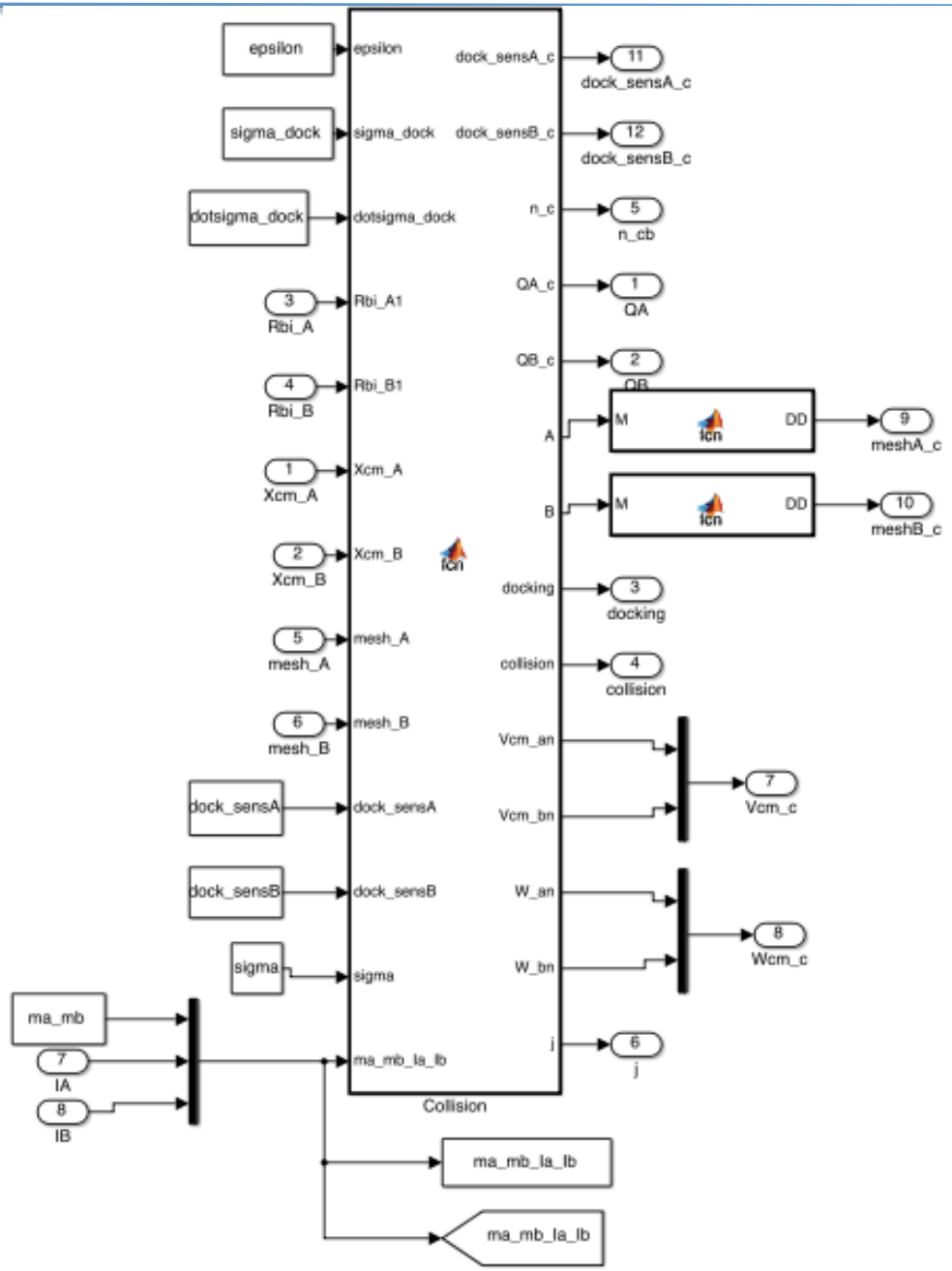




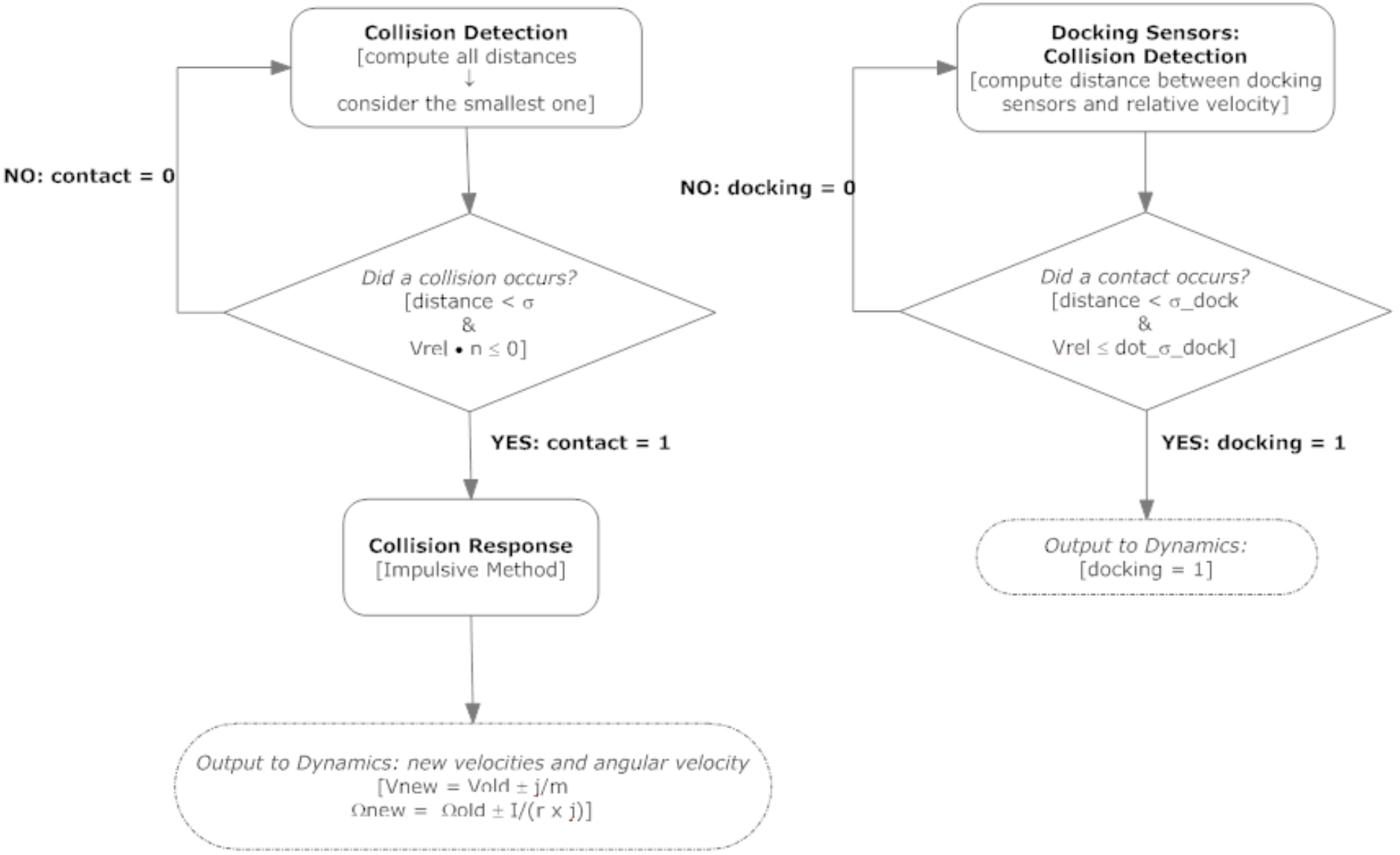


Simulink Collision Block: Parameter Mask





Simulink Collision Block: Parameter Mask



Simulink Collision Block: Inputs

- Two vectors representing the states (position, velocity and angular velocity) of the two bodies. These vectors needs to be expressed in an INERTIAL RF as:

$$Xcm_A = \begin{bmatrix} x_A & y_A & z_A & v_{xA} & v_{yA} & v_{zA} & \omega_{xA} & \omega_{yA} & \omega_{zA} \end{bmatrix}^T$$

$$Xcm_B = \begin{bmatrix} x_B & y_B & z_B & v_{xB} & v_{yB} & v_{zB} & \omega_{xB} & \omega_{yB} & \omega_{zB} \end{bmatrix}^T$$

- Two rotational matrices, from BODY RF to INERTIAL RF, for the two bodies. As an example:

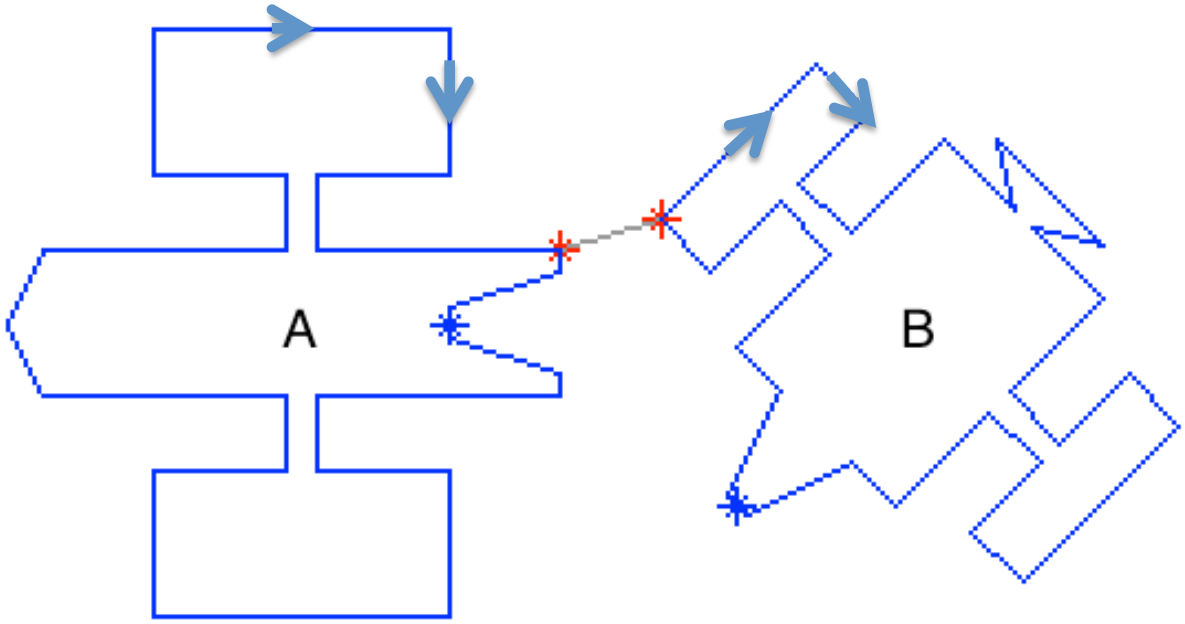
$$R_{ib}^A = \begin{bmatrix} \cos(\theta_A) & \sin(\theta_A) \\ -\sin(\theta_A) & \cos(\theta_A) \end{bmatrix}, \; R_{ib}^B = \begin{bmatrix} \cos(\theta_B) & \sin(\theta_B) \\ -\sin(\theta_B) & \cos(\theta_B) \end{bmatrix}$$

Simulink Collision Block: Inputs

- n points on body A and m points on body B, expressed in the BODY RF, as respectively a (3,n+1) and a (3,m+1) matrices:

$$mesh\_A = \begin{bmatrix} P_{x1} & P_{x2} & \dots & P_{xn} \\ P_{y1} & P_{y2} & \dots & P_{yn} \\ P_{z1} & P_{z2} & \dots & P_{zn} \end{bmatrix}, \quad mesh\_B = \begin{bmatrix} P_{x1} & P_{x2} & \dots & P_{xm} \\ P_{y1} & P_{y2} & \dots & P_{ym} \\ P_{z1} & P_{z2} & \dots & P_{zm} \end{bmatrix}$$

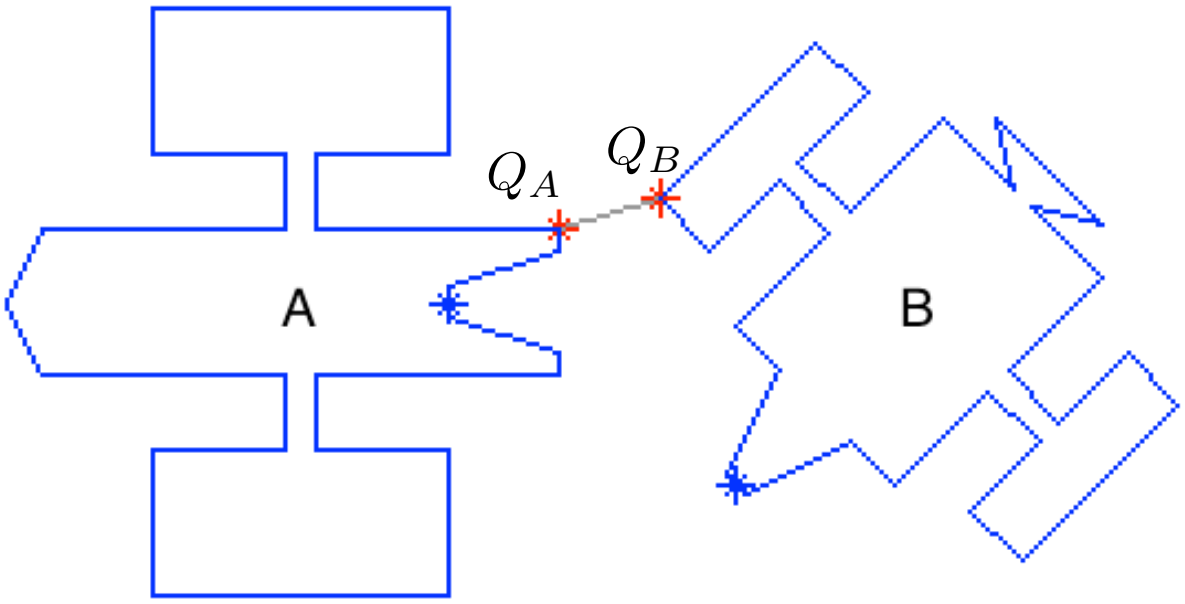
- The vertices of the bodies have to be declared *clockwise*.



Simulink Collision Block: Outputs

- The two closest points one on each body, expressed in INERTIAL RF as:

$$Q_A = \begin{bmatrix} x \\ y \end{bmatrix}_A, \; Q_B = \begin{bmatrix} x \\ y \end{bmatrix}_B$$



Simulink Collision Block: Outputs

- Two variables that indicate respectively:

- ❑ $collision$ this variable is initialized to be 0 and becomes 1 when any *collision* between the two bodies occurs;

- ❑ $docking$ this variable is initialized to be 0 and becomes 1 when the *docking* between the two bodies occurs = two sensors are touching each other;

- Normal to the collision surface of the second body B (convention):

$$\boldsymbol{n_{cB}}$$

- Because we are not considering *friction*, the impulse generated by a collision is just the normal impulse:

$$\boldsymbol{j} = j\boldsymbol{n_{cB}}$$

Simulink Collision Block: Outputs

- n points on body A and m points on body B, expressed in the INERTIAL RF, propagated as the simulation is running. They are two vectors in the form:

$$meshA\_c = [P_{x1}\ P_{y1}\ P_{x2}\ P_{y2}\ \ldots\ P_{xn}\ P_{yn}]$$

$$meshB\_c = [P_{x1}\ P_{y1}\ P_{x2}\ P_{y2}\ \ldots\ P_{xm}\ P_{ym}]$$

- Sensors positions, expressed in the INERTIAL RF, propagated as the simulation is running. They are two vectors in the form:

$$dock\_sensA\_c = [x\ y]^T$$

$$dock\_sensB\_c = [x\ y]^T$$

- Velocity and Angular velocity of the centers of mass of the bodies, expressed in INERTIAL RF, propagated as the simulation is running and expressed as:

$$V_{cm}^c = [v_{xA}\ v_{yA}\ v_{zA}\ v_{xB}\ v_{yB}\ v_{zB}]^T$$
$$\Omega_{cm}^c = [\omega_{xA}\ \omega_{yA}\ \omega_{zA}\ \omega_{xB}\ \omega_{yB}\ \omega_{zB}]^T$$

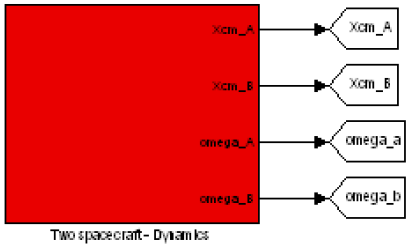
Simulink Collision Block: Outputs

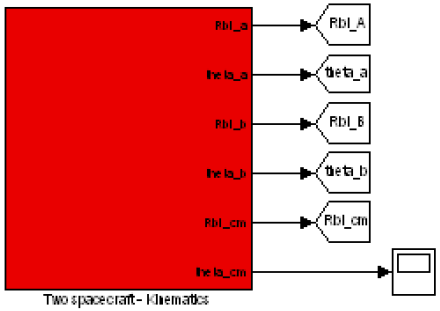
- Linear velocities and angular velocities for body A and B after the impact

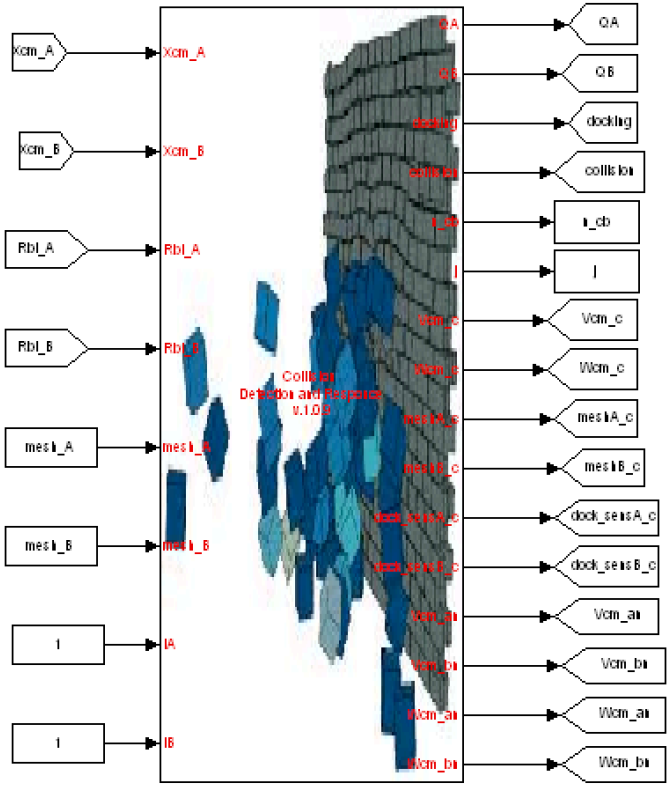
$$Vcm_{An} = [V_{xA}\ V_{yA}\ 0]^T$$
$$Vcm_{Bn} = [V_{xB}\ V_{yB}\ 0]^T$$

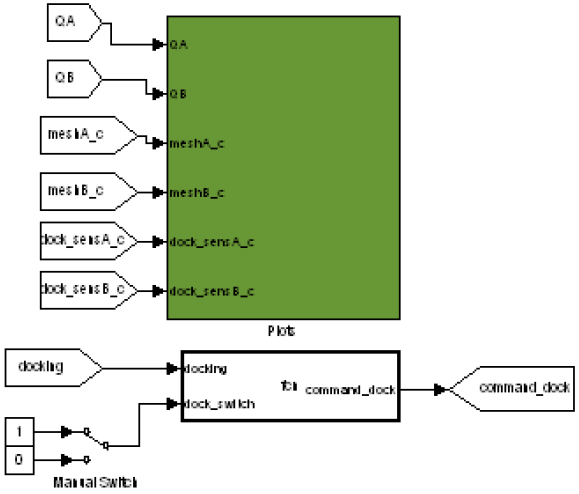
$$\Omega cm_{An} = [\omega_{xA}\ \omega_{yA}\ \omega_{zA}]^T$$
$$\Omega cm_{Bn} = [\omega_{xB}\ \omega_{yB}\ \omega_{zB}]^T$$

Modeling Collision





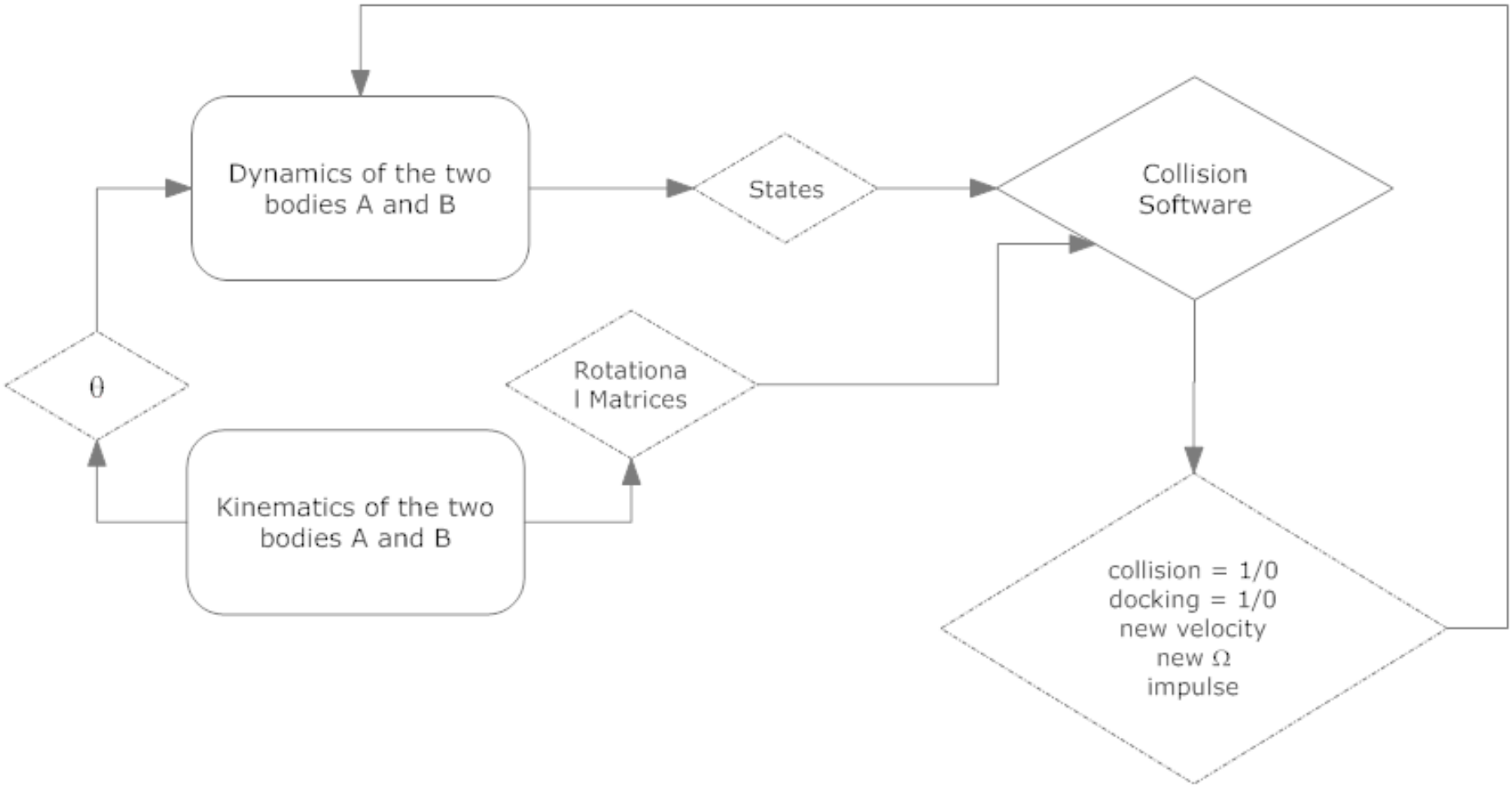




INPUTS:
- Xcm_A and Xcm_B: vectors with position, velocity and angular velocity (in INERTIAL RF) of the object A and B in the form:
  Xcm_A = [xA yA zA vxA vyA vzA wxA wyA wzA]';
- Rbi_A and Rbi_B: rotational matrices from BODY to INERTIA of both bodies A and B (2D)
- mesh_A and mesh_B: n points on each body in BODY RF (fixed in the center of mass of the sc) as a (3, n) matrix in the form:
  mesh_A = [Px1 Px2 .... Pxn;
  Py1 Py2 .... Pxn;
  Pz1 Pz2 .... Pzn];
- IA and IB: inertia of each body w.r.t its CM around z-axis;

OUTPUTS:
- Q_A and Q_B: positions of the two contact points respectively on the two bodys as,
  Q_A = [xQA yQA]';
- docking: docking = 0 and become 1 when the docking heppens between sensor_A and sensor_B;
- collision: collision = 0 and become 1 when a collision heppens between the points QA and QB;
- n_cb: normal to the collision surface of the B-body;
- j: impuse generated by the collision;
- Vcm_c: velocities of the center of mass of both bodies after collision as:
  Vcm_c = [vxA vyA 0 vxB vyB 0]';
- Wcm_c: angular velocities of the center of mass of both bodies after collision as:
  Wcm_c = [wxA wyA 0 wxB wyB 0]'.
- meshA_c and meshB_c: n points on each body in INERTIAL RF propagated during the simulation, as a (1, 3*n) vector in the form:
  pointsA_c = [Px1 Py1 Px2 Py2 .... Pxn Pyn];
- dock_sensA_c and dock_sensB_c: position of the docking sensors on A and B in INERTIAL RF while simulation is running;
- Vcm_an and Vcm_bn: velocities of A and B after dockin as:
  Vcm_An = [vxA vyA 0]'
- Wcm_an and Wcm_bn: angular velocities of body A and B:
  Wcm_an = [0 0 wzA]'s

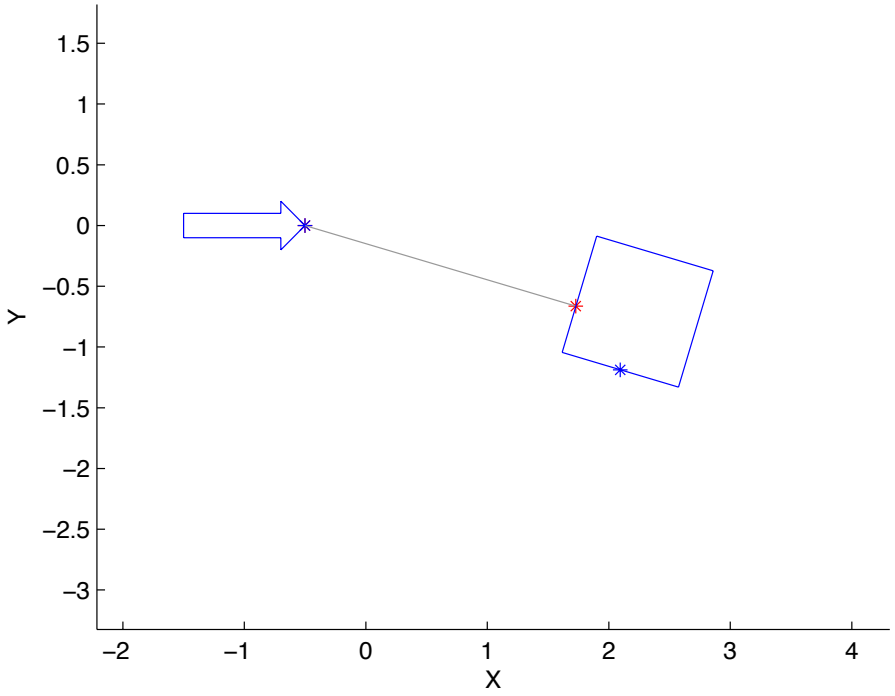
Modeling Collision



Modeling Collision

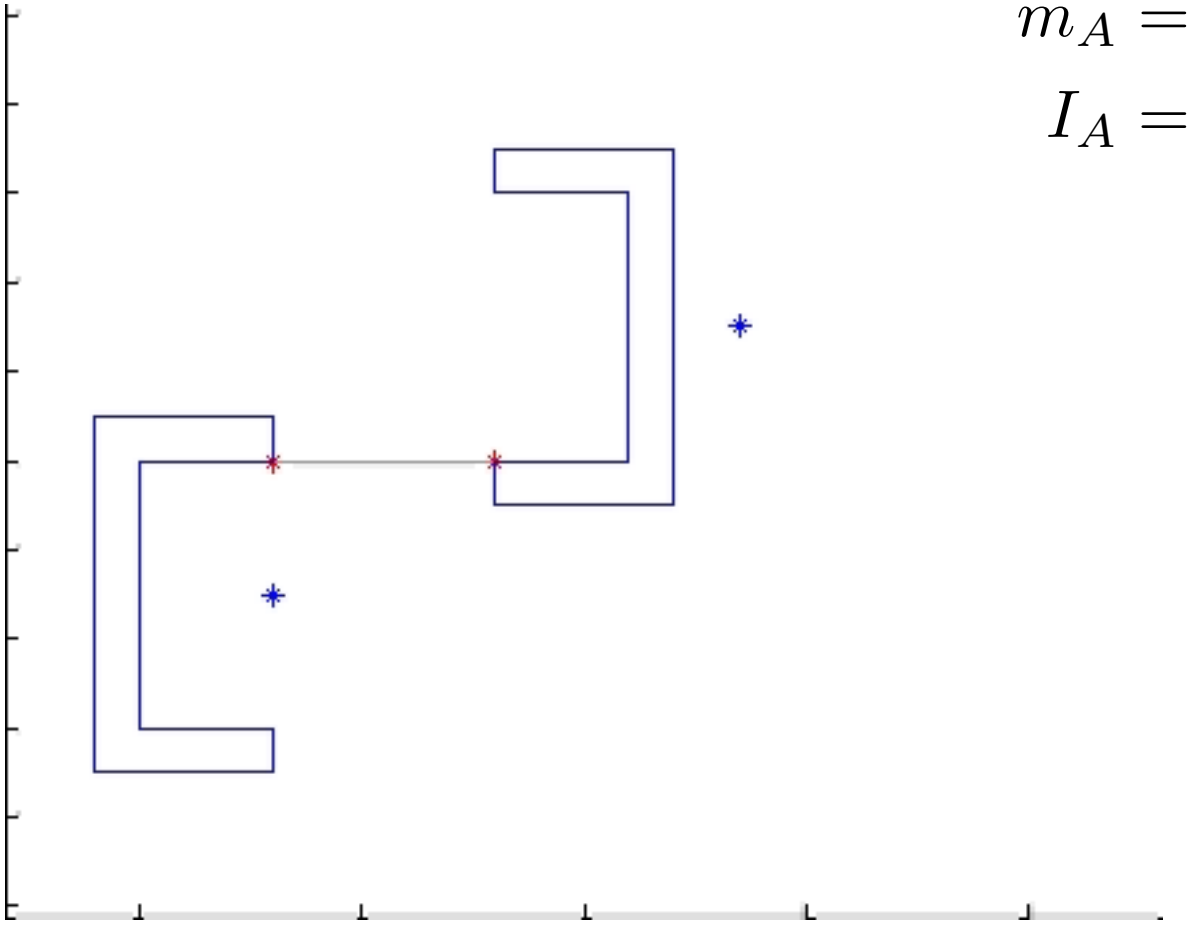
- Simplified Case: we assume that the manipulator has infinity mass and inertia wrt the target (IA = 1e6 and mA = 1e6 (only for collision!!)). In this way you can avoid to propagate back the collision response along the manipulator's joints;

- We assume that the manipulator is rigid body for each time sample of the collision: this makes sense since the motion of manipulator is much slower then the collision block sample time;



THANK YOU

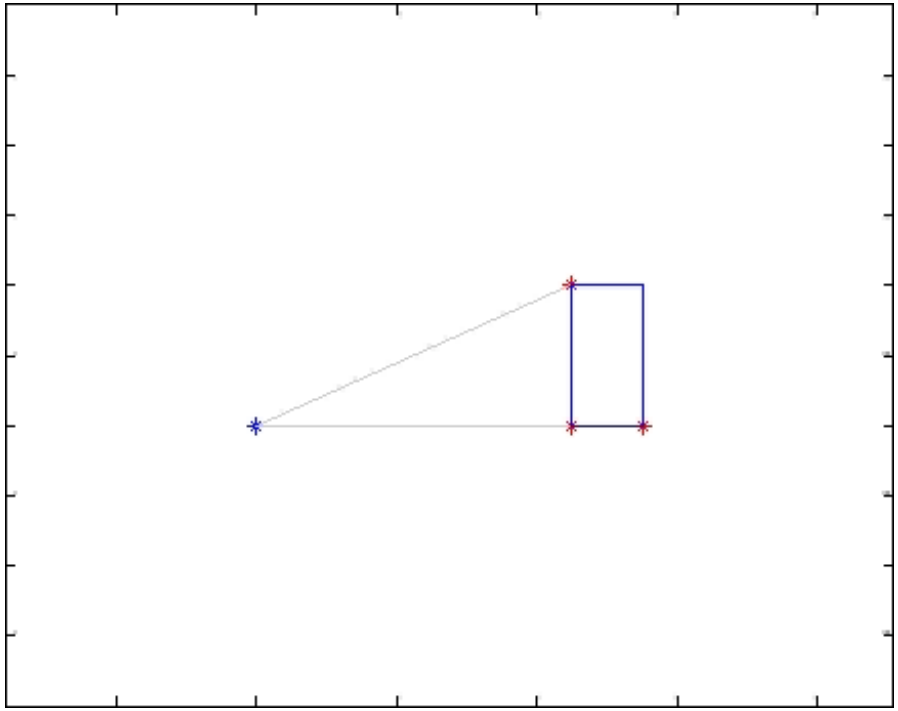
Testing Conservation of Energy and Momentum

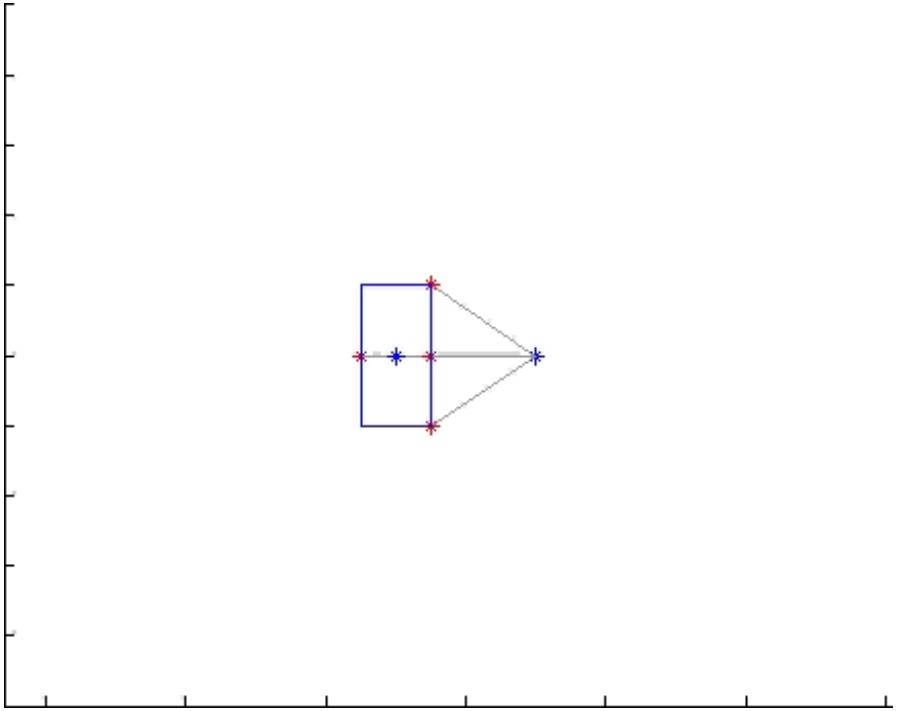


$$m_A = m_B$$

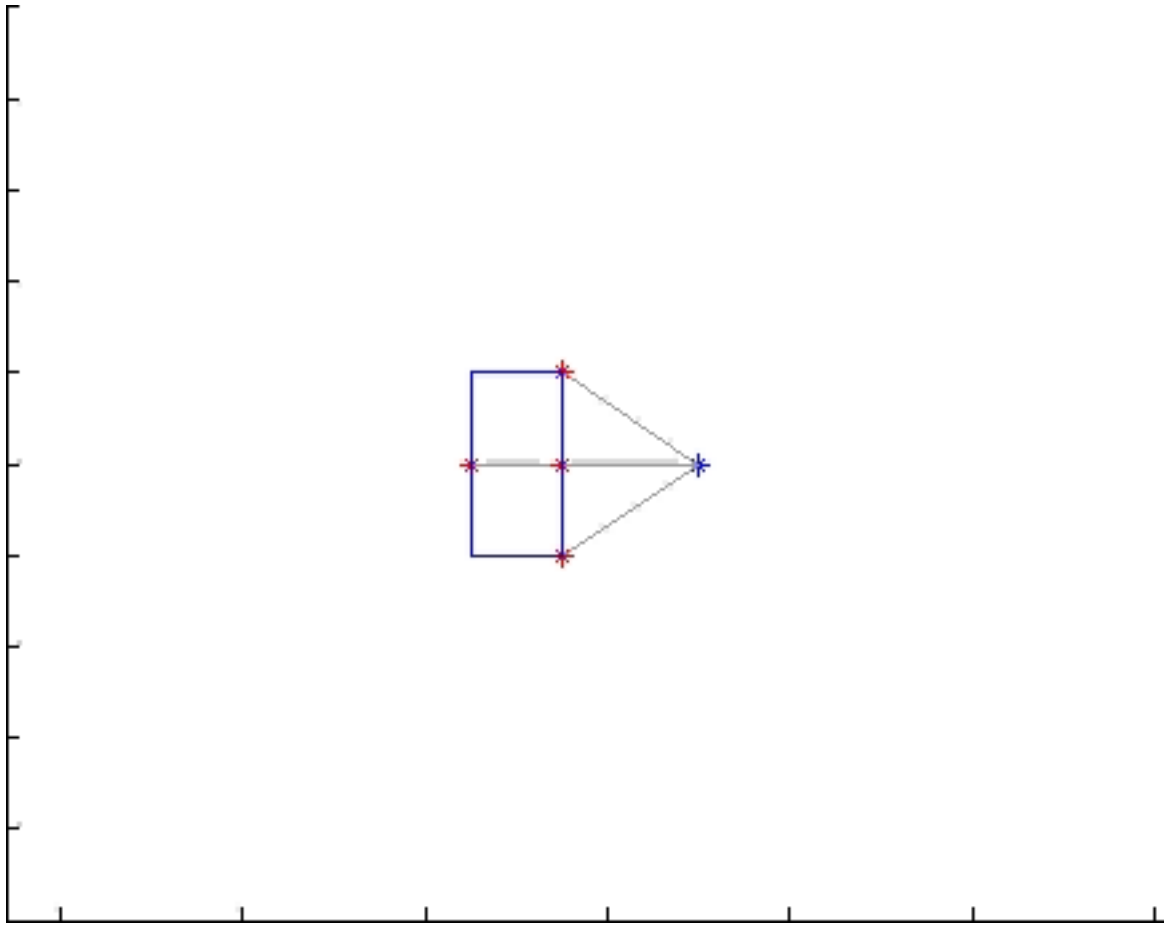
$$I_A = I_B$$

Collision Detection: Point to Polygon

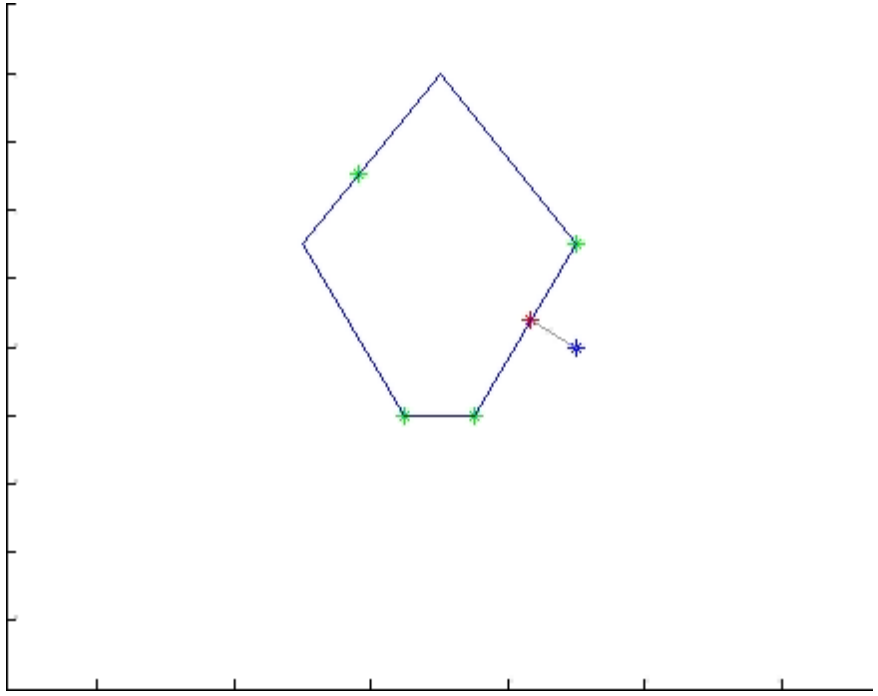




Collision Detection: Point to Polygon: Dynamics

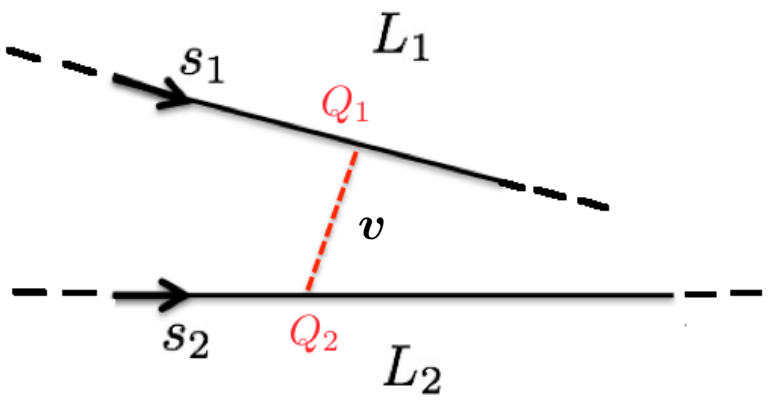


Point to Polygon: Minimum Distance



Collision Detection: Two Lines

- Consider L1 and L2 as infinite lines:



If we define the vector v as the distance between this two points:

$$\boldsymbol{v}(t_{d1}, t_{d2}) = L(t_{d1}) - L(t_{d2})$$

This vector must be perpendicular to both lines.

- Therefore:

$$L_1(t) \cdot \boldsymbol{v} = 0$$
$$L_2(t) \cdot \boldsymbol{v} = 0$$

- After algebraic manipulations:

$$t_{d1} = \frac{(bf - ce)}{ae - b^2}$$

$$t_{d2} = \frac{f - bc}{ea - b^2}$$

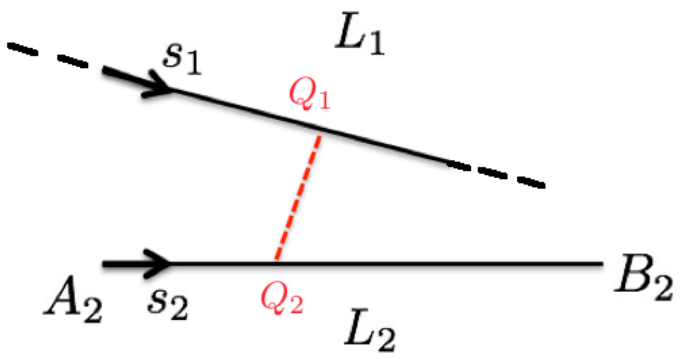
$$a = (B_1 - A_1) \cdot (B_1 - A_1)$$
$$b = (B_1 - A_1) \cdot (B_2 - A_2)$$
$$c = (B_1 - A_1) \cdot (A_1 - A_2)$$
$$e = (B_2 - A_2) \cdot (B_2 - A_2)$$
$$f = (B_2 - A_2) \cdot (A_1 - A_2)$$

Collision Detection: Two Line Segments

- Notice that: *only when both happen to points lie on the segments does this expressions apply.*
  - ➢ A common misconception: *it is sufficient to clamp t points to the nearest endpoint.*

- In more general cases we use those expressions as IC, as:

1. Consider L1 as an infinite line and calculate the closest point to Q2 (generic point on L2) on L1:



$$t_{d1} = \frac{(bf - ce)}{ae - b^2}$$

2. Then evaluate:

$$t_{d2} = \frac{bt_{d1} + f}{e}$$

and:

$$if \quad t_{d2} < 0 \quad \rightarrow \quad t_{d2} = 0 \quad \rightarrow \quad t_{d1} = \frac{bt_{d2} - c}{a}$$

$$if \quad t_{d2} > 1 \quad \rightarrow \quad t_{d2} = 1 \quad \rightarrow \quad t_{d1} = \frac{bt_{d2} - c}{a}$$