Real-time Physics Engine

김영준
이화여대
kimy@ewha.ac.kr

김진욱
KIST
jwkim@imrc.kist.re.kr

Physics Simulation in Computer Games

- Simulate the physical behavior of gaming objects in real-time

Rigid body

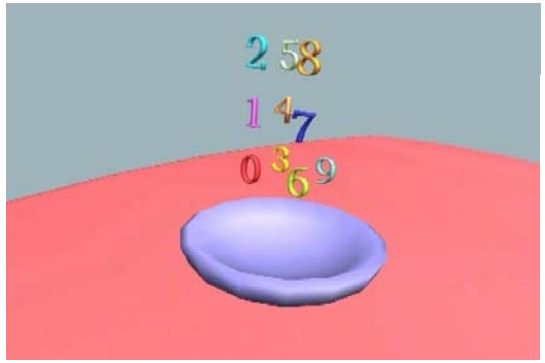
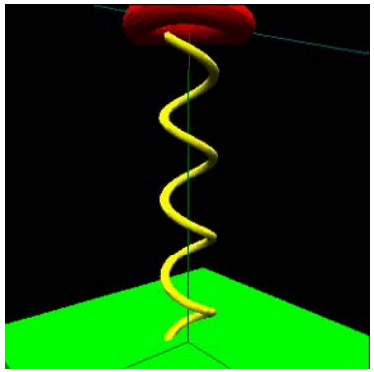
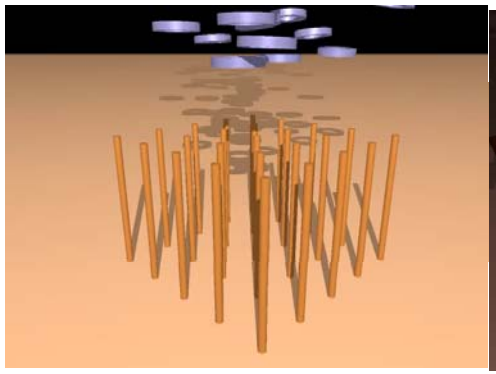
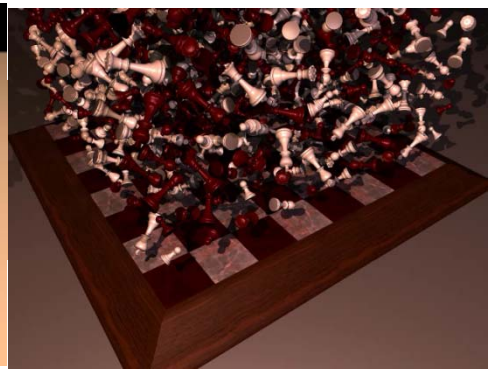
Articulated body (ragdoll)

Particle

Vehicle

Lecture Scope

- Focus on rigid body dynamics
  - Basic component of physics engines
  - Non-trivial to implement due to many sub-components being required
  - Most of hard part can be downloadable (e.g., collision detection, ODE solver)

Target Audience

- Intermediate
  - Know a bit and want to implement physics engine by myself
  - Basic math, physics background

Buzz Worthy Messages

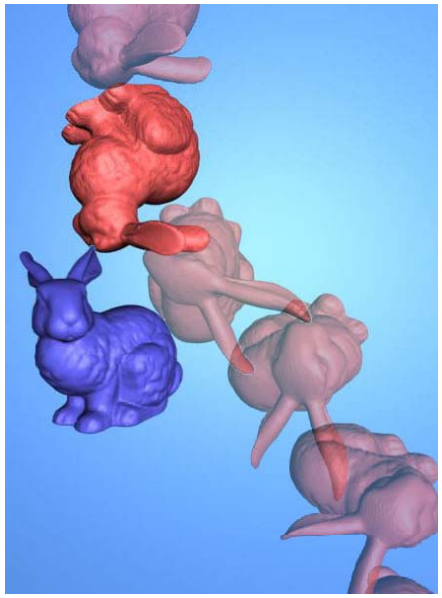
- Governing equations
  - Newton/Euler, linear/angular momentum, intertia tensor
- Basic ODE solvers
  - Solve $\dot{\mathbf{x}} = f(\mathbf{x}, t)$
  - Euler, mid-point, runge-kutta
- Collision detection
  - CD, separation distance, penetration depth
- Collision response
  - Impulse, LCP (linear complementarity problem)

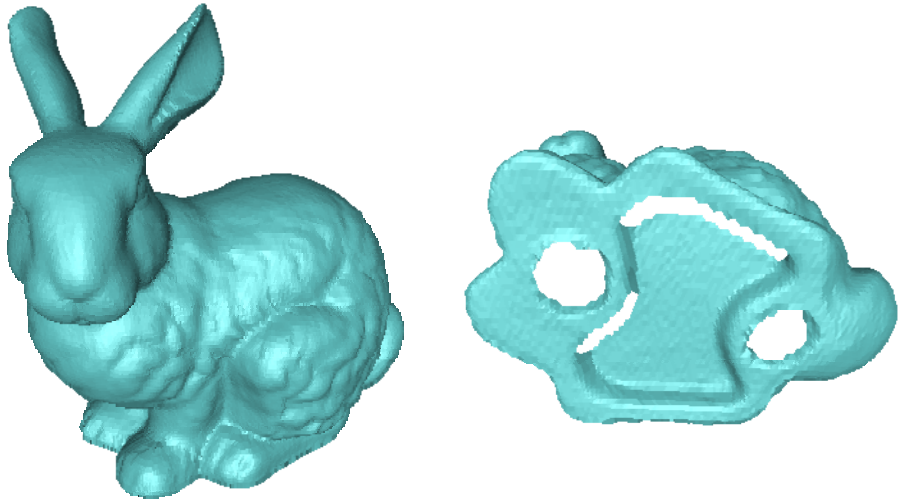
Schedule

1. General introduction to rigid body dynamics (김영준, 15 mins)
2. Unconstrained rigid body dynamics (김진욱, 40 mins)
3. ODE basics (김영준, 20 mins)
4. Break (5 mins)
5. Collision detection (김영준, 1 hr)
6. Contact dynamics (김진욱, 40 mins)

Recent Trends - Research

- Generality
  - Geometry (e.g. non-convex)
  - Topology (e.g. polygon soups, point-set)

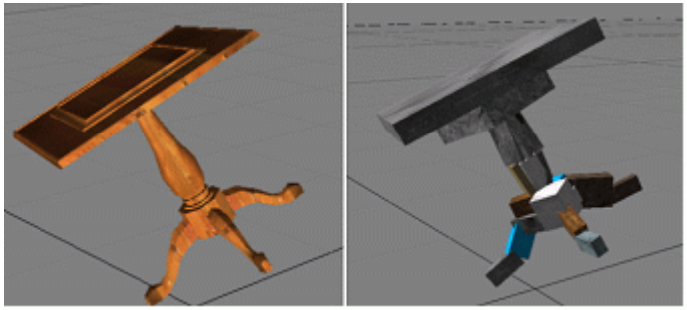




Recent Trends - Research

- Scalability
  - Many bodies
  - Simulation approximation (e.g. simulation LOD, approximate convex decomposition)

Adaptive dynamics of articulated characters



- Stability
  - Scenario (e.g. stacking, friction)
  - Stiff system

Recent Trends - Practice

- Next-gen hardware platform + parallelism
  - Multi-core CPUs, SPUs (Cell)
  - GPUs
  - PPUs (probably gone)
  - Intel buys Havok (2007), nVIDIA buys Ageia (2008)
- Specialized, high-performance physics engines still prevail
  - E.g. Natural motion (character), Pixelux (fracture)
  - E.g. idSoft, Crytek (customized)

NEXT STOP: UNCONSTRAINED DYNAMICS

- Slides will be available:

http://graphics.ewha.ac.kr

HCI2008 Tutorial

실시간게임 물리엔진기술

Rigid Body Dynamics

KIST Imaging Media Research Center

김진욱

Physics in Game

- Realtime
  - at least 30 frames/second
  - should not consume too much resources.
- Plausibleness
  - It's enough to look natural.
  - Approximation may work.
- Interactivity
  - different from pre-defined animation.
  - should interact with users and other game things.

Overview

- **Particle Dynamics**
- Rigid Body Dynamics
- Collision and Contact
- Constraint/Joint
- Introduction to VirtualPhysics

Particle Dynamics

- Idealized body without volume
- Newton's law

$$f = m\ddot{x}$$

- 2nd order ordinary differential equation
- 1st order ODE will be easier

Particle Dynamics

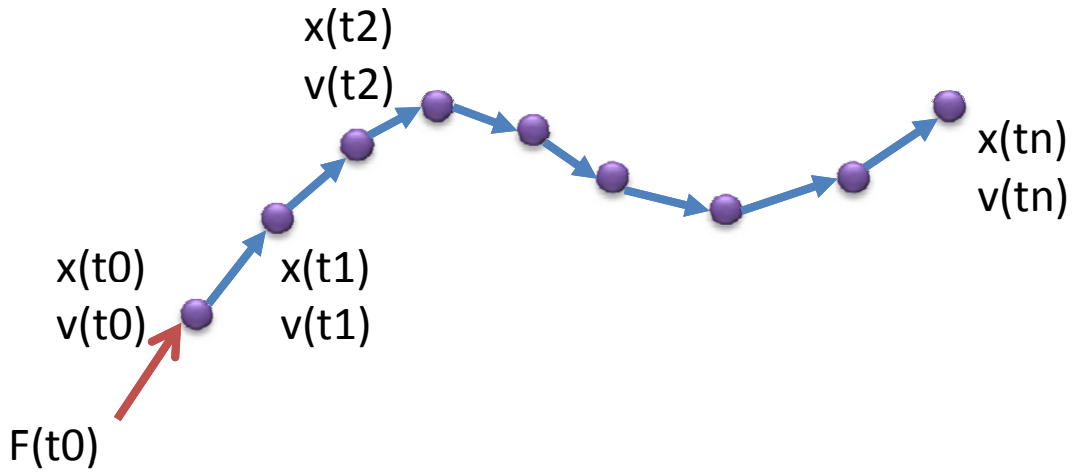
- Let us introduce a new variable $v = \dot{x}$
- Then $\frac{dv}{dt} = f / m$
- Now we can represent the Newton's physics law as 1st order O.D.E

$$\frac{d}{dt}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} v \\ f / m \end{pmatrix}$$

- Dynamics state : position, velocity

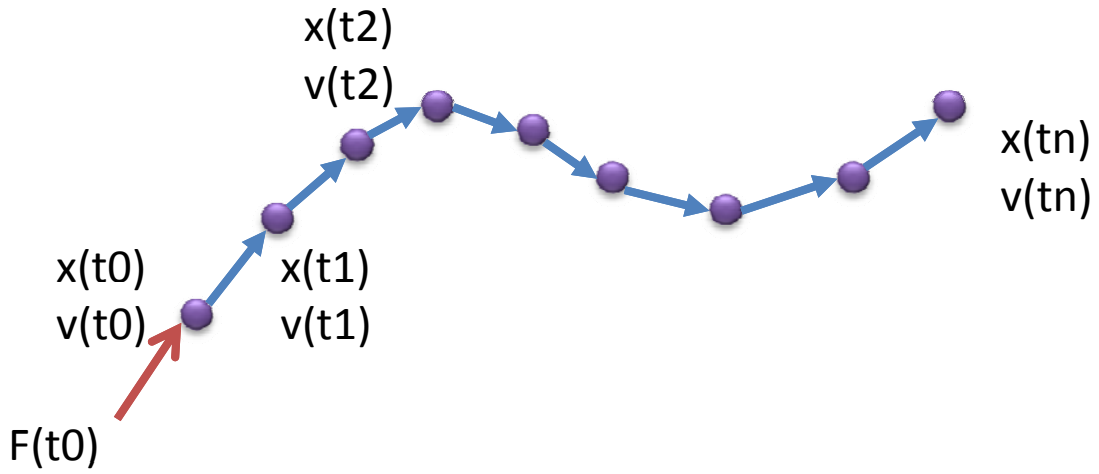
Typical Problems of Dynamics

- Forward Dynamics
  - Given the force applied to the particle, find the acceleration of the particle.
  - In some cases, all the forces are not provided. Instead, we have constraint equations.
  - Ex: Simulation



Typical Problems of Dynamics

- Inverse Dynamics
  - Given the acceleration of the particle,
    find the force that drives the particle
  - We may have only the trajectory of the particle
  - Ex: Robot motion planning



Forward Dynamics of Particle System

- Evaluate force applied to the particle
  - gravity, drag, spring, damper, collision/contact, …

$$f = mg - k_d v + k_s (x - x_i) + \cdots$$

- Calculate acceleration

$$a = f / m$$

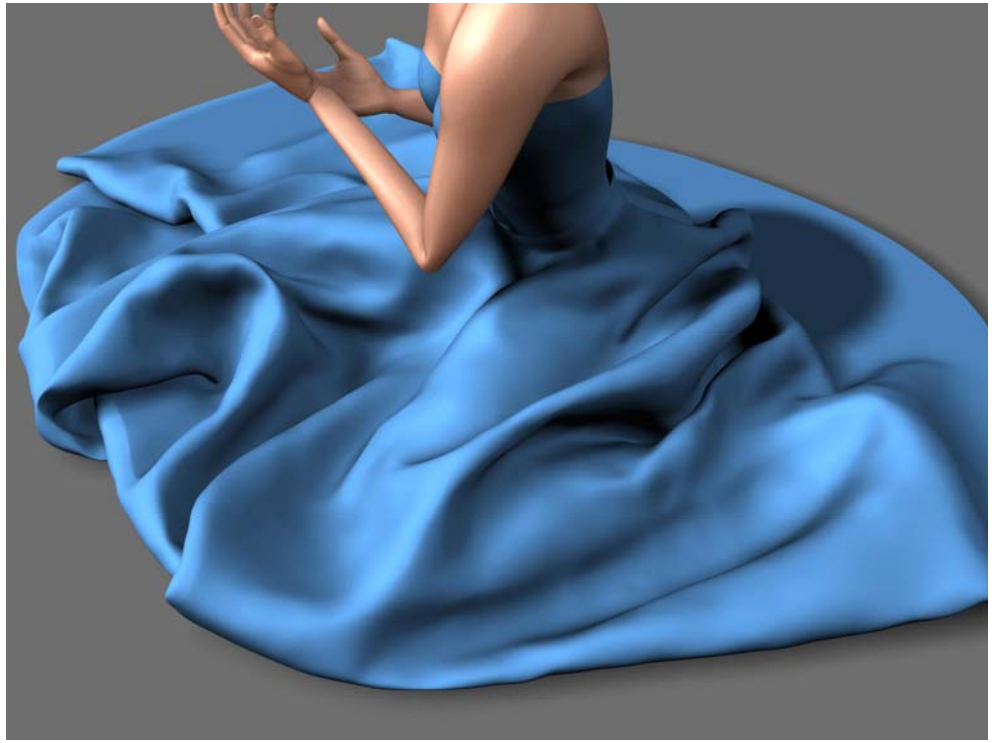
- Step forward in time
  - Integrate dynamics

$$x(t_{n+1}) = x(t_n) + \Delta t \; v(t_n)$$

$$v(t_{n+1}) = v(t_n) + \Delta t \; a(t_n)$$

Examples of Particle System

Hair simulation
H. Halen
Digital Illusions



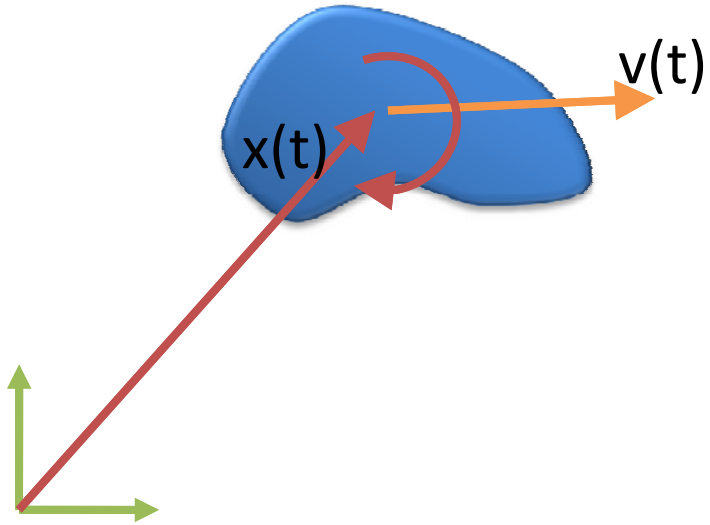
Cloth simulation
N. Govindaraju et.al
UNC

Overview

- Particle Dynamics
- **Rigid Body Dynamics**
- Collision and Contact
- Constraint/Joint
- Introduction to VirtualPhysics

Rigid Body

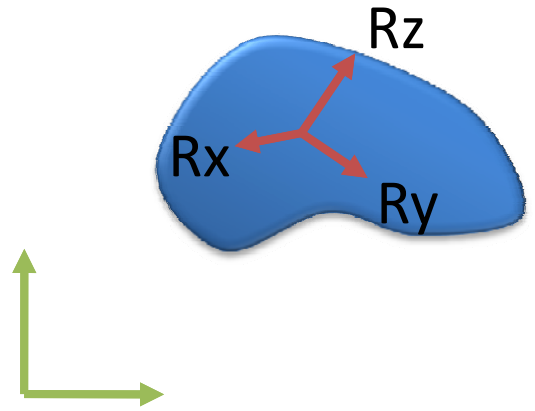
- Now a body has a volume
- Dynamics state
  - not only position and velocity but also ?



Rotation matrix

- How to represent the orientation of the rigid body?
- Attach a coordinate frame to the body
  - Rx: a column vector representing X-axis of the frame
  - Ry: a column vector representing Y-axis of the frame
  - Rz: a column vector representing Z-axis of the frame
- Build a rotation matrix

$$R = \begin{bmatrix} | & | & | \\ R_x & R_y & R_z \\ | & | & | \end{bmatrix}$$



Something you should know about Rotation Matrix

- Useful properties
  - Orthogonality

$$RR^T = R^T R = I$$

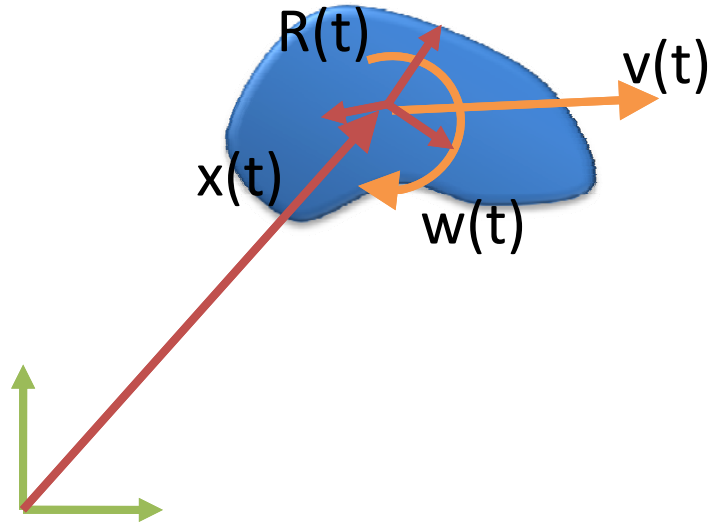
  - Time derivative

$$\dot{R}R^T + \left(\dot{R}R^T\right)^T = 0$$

- Parameterization is evil
  - Euler angle, quaternions, exponential map
  - All the parameterization will fail.

State of Rigid Body

- Dynamics State
  - position(x), linear velocity(v)
  - orientation(R), angular velocity(w)



Angular Velocity

- Angular velocity

$$w = [w_x \quad w_y \quad w_z]^T$$

- Relation

$$\dot{R} = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} R$$

Rigid Body Dynamics

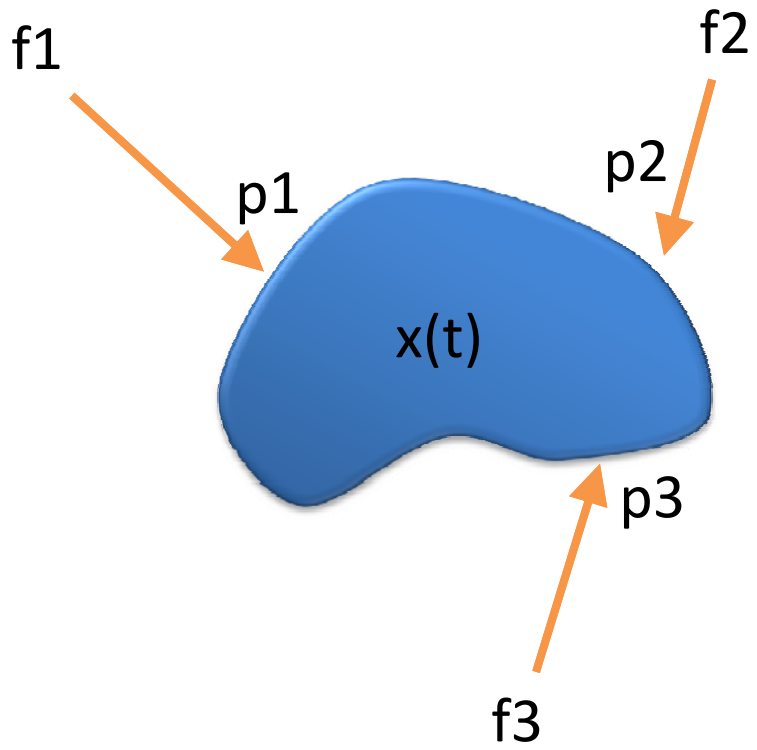
- Newton's law

$$f = m\dot{v}$$

$$\tau = I\dot{w} + w \times (Iw)$$

- (Moment of) Inertia tensor
  - 3X3 positive definite symmetric matrix
  - Think of a bicycle wheel

Net Force and Torque



$$f = \sum f_i$$

$$\tau = \sum \left(p_i - x(t) \right) \times f_i$$

Integrating Rigid Body Dynamics

- Recall particle dynamics

$$x(t_{n+1}) = x(t_n) + \Delta t\ v(t_n)$$
$$v(t_{n+1}) = v(t_n) + \Delta t\ a(t_n)$$

- Then, is it OK?

$$R(t_{n+1}) = R(t_n) + \Delta t\ w(t_n)$$
$$w(t_{n+1}) = w(t_n) + \Delta t\ \alpha(t_n)$$

- Unfortunately the rotation matrix is not in vector space!
  - Addition or scalar multiplication is not available.
  - Fortunately the angular velocity is a vector.

Integrating Rotation Matrix

- Use quaternions

$$q(t_{n+1}) = q(t_n) + \Delta t\ \dot{q}(t_n)$$

$$\dot{q}(t) = [0,\ w(t)/2]\, q(t)$$

- Exponential map

$$R(t_{n+1}) = R(t_n) e^{\Delta t[w(t_n)]}$$

- Or simply

$$R(t_{n+1}) = R(t_n)\left(I + \Delta t[w(t_n)]\right)$$

  - You may need to normalize the matrix.

Overview

- Particle Dynamics
- Rigid Body Dynamics
- **Collision and Contact**
- Constraint/Joint
- Introduction to VirtualPhysics

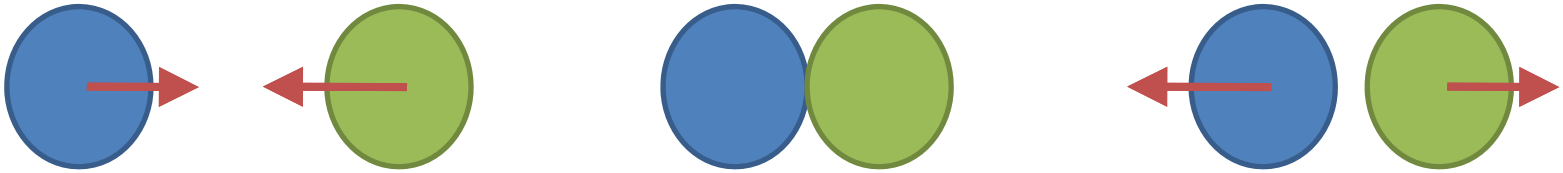
Collision & Contact

- Collision
  - A body hits another body.
  - Relative normal velocity is negative.
  - ***Impulse*** governs the dynamics.

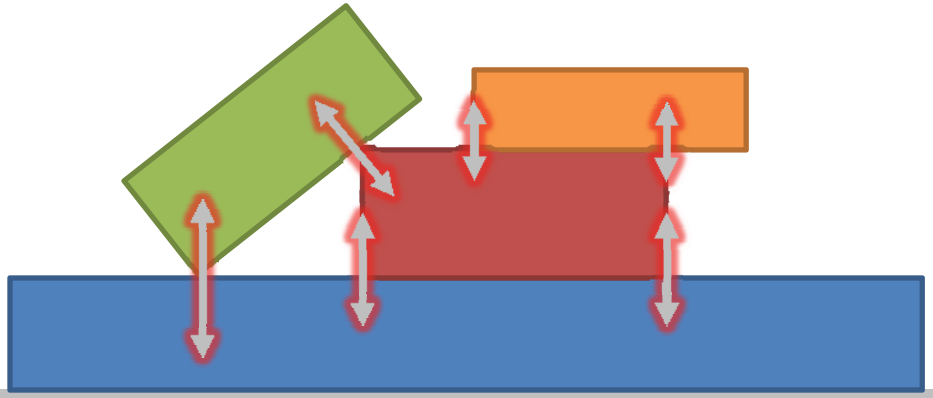
- Contact
  - A body rests against another body.
  - Relative normal velocity is negligible.
  - ***Resting force*** prevents inter-penetration.

Collision & Contact

- Collision



- Contact

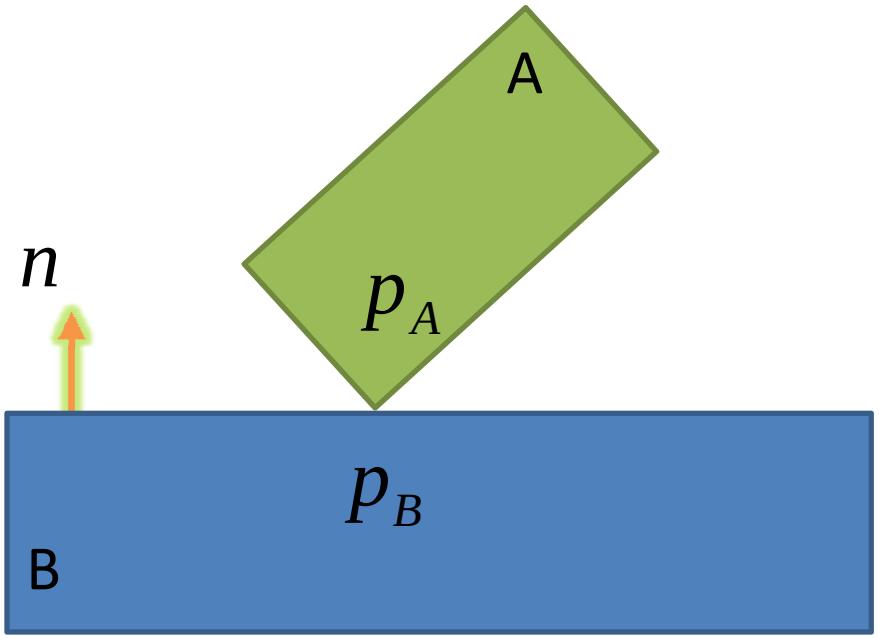


Collision and Impulse

- Collision

$$p_A = p_B$$

$$n \cdot (\dot{p}_A - \dot{p}_B) < 0$$



Collision and Impulse

- Impulse will alter the velocity of bodies instantly.

$$n \cdot (\dot{p}_A^+ - \dot{p}_B^+) = -\varepsilon\, n \cdot (\dot{p}_A^- - \dot{p}_B^-)$$

$\dot{p}^+$: velocity right after the collision
$\dot{p}^-$: velocity right before the collision
ε : coefficient of restitution

Impulse Dynamics

$$j = m\Delta v$$

$$r \times j = I\Delta w$$

- Calculate impulse and altered velocity

$$j = \frac{-(1+\varepsilon)v_{rel}^{-}}{m_a^{-1} + m_b^{-1} + n \cdot \left(I_a^{-1}(r_a \times n)\right) \times r_a + n \cdot \left(I_b^{-1}(r_b \times n)\right) \times r_b}$$

$$v_a^{+} = v_a^{-} + \frac{j}{m_a} n \quad , \quad w_a^{+} = w_a^{-} + I_a^{-1}(r_a \times jn)$$

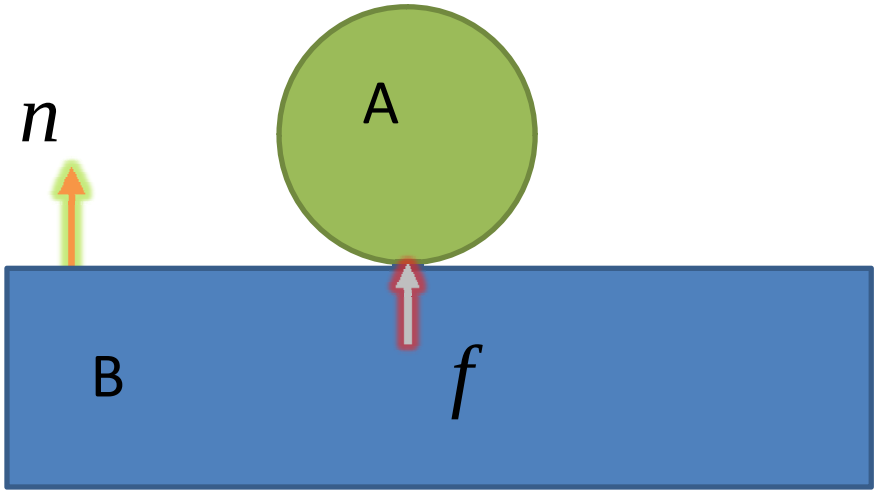
Contact and Resting Force

- Contact

$$n \cdot \ddot{p}_A = a_{rel} \geq 0$$

or

$$cf + b \geq 0$$

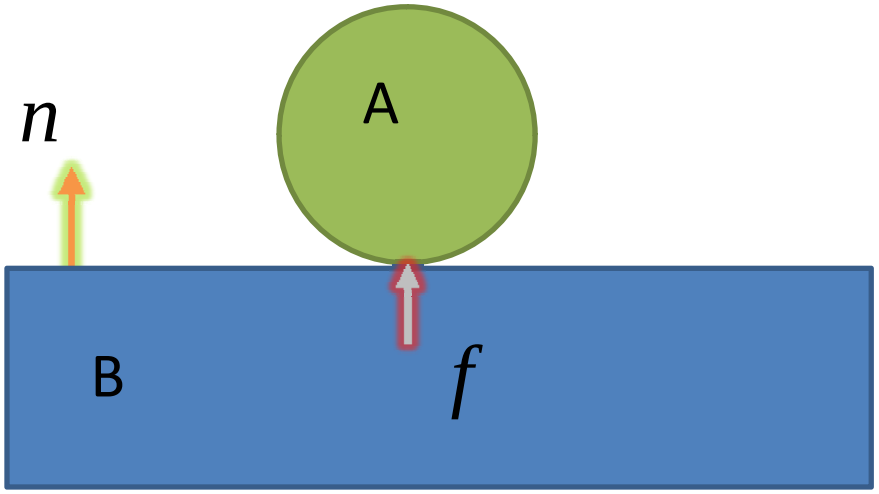


Two Possible Conditions

$$cf + b > 0 \qquad cf + b = 0$$

$$f = 0 \quad \text{or} \quad f \geq 0$$

Non-penetration repulsive force



Linear Complementary Problem

$$0 \leq f \perp cf + b \geq 0$$

- Algebraic approach
  - Lemke's method
  - Dantzig's method
- Iterative approach
  - Projective Gauss-Seidel method

Things we didn't count for

- Friction
- Multiple collision/contact
- Refer to
  - D. Baraff. Fast contact force computation for nonpenetrating rigid bodies., *SIGGRAPH* 1994
  - Chris Hecker http://www.chrishecker.com/Rigid_Body_Dynamics
  - Erin Catto. http://www.gphysics.com
  - Forum on Bullet physics library http://www.bulletphysics.com/Bullet/phpBB3/viewforum.php?f=4

Overview

- Particle Dynamics
- Rigid Body Dynamics
- Collision and Contact
- **Constraint/Joint**
- Introduction to VirtualPhysics

Degrees of Freedom

- There are ***2 DOF*** associated to the motion of a particle in 2D space.
  - DOF = 2(position)
- There are ***3 DOF*** associated to the motion of a rigid body in 2D space.
  - DOF = 2(position) + 1(orientation)
- There are ***6 DOF*** associated to the motion of a rigid body in 3D space.
  - DOF = 3(position)+3(orientation)

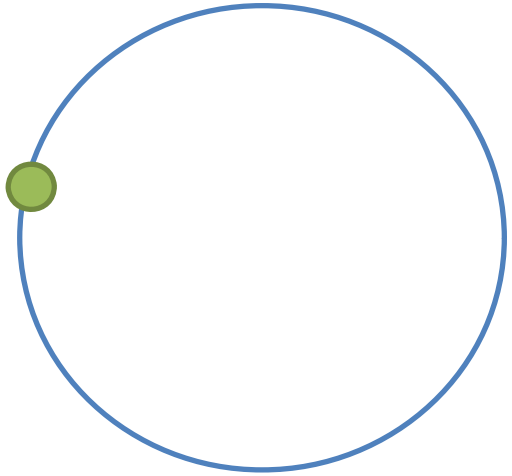
Constraint

- You can make ***almost anything*** with particles and springs as long as it's jello.
- ***Constraints*** will give you ***rigid links*** between bodies.
- Penalty method
  - Naïve approach to approximate constraints
  - Easy to implement
  - Painful to find right parameters
  - Tend to be numerically unstable

A Point on a Circle

- Number of variables = 2

$$p = (x, y)$$



- Number of constraints = 1

$$\| p \| - r = 0$$

- Number of independent variables= 2-1=1

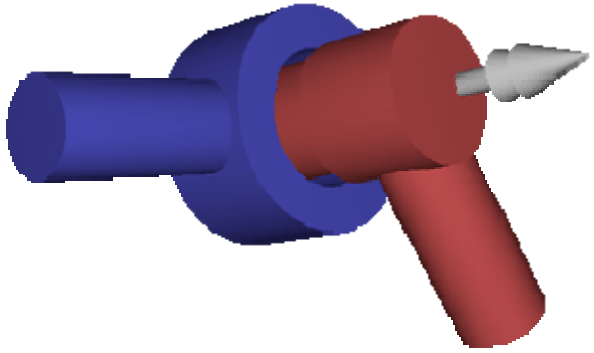
$$x = r\cos\theta\ ,\ y = r\sin\theta$$

Handling Constraints

- Lagrangian approach
  - Solve dynamics with implicit constraint equations
  - In most of the available physics engine
  - $O(n^3)$
- Independent coordinates approach
  - Express dynamics only with the independent varables
  - Popular in Robotics
  - $O(n^3)$: Composite body algorithm
  - O(n) : Featherstone algorithm

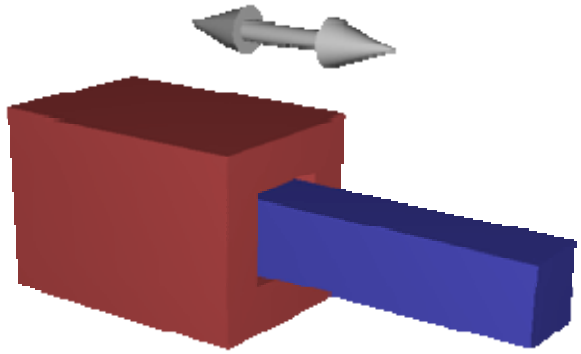
Rotational Joint

- Connected bodies can rotate relatively about a rotational axis.
- Constrains 5 DOF motion
- DOF of the Joint = 1
- Ex: Door Hinge



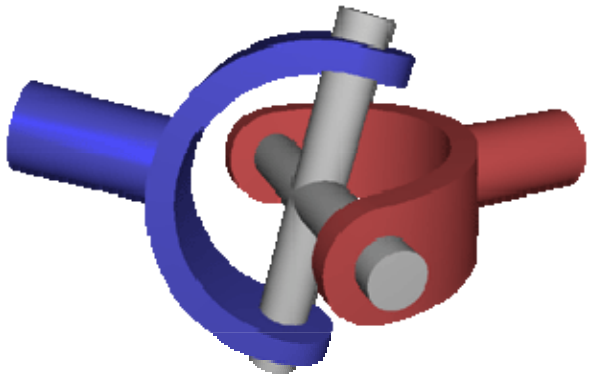
Prismatic Joint

- Connected bodies can slide relatively about a given direction.
- Constrains 5 DOF motion
- DOF of the Joint = 1



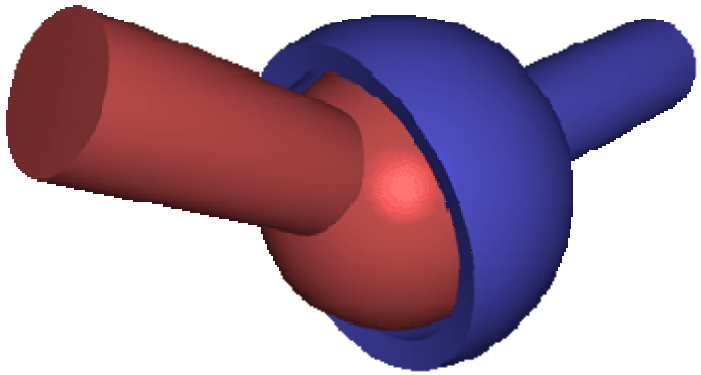
Universal Joint

- Connected bodies can rotate relatively about two consecutively attached rotational axes.
- Constrains 4 DOF motion
- DOF of the Joint = 2



Ball Joint

- Connected bodies can rotate freely.
- Constrains 3 DOF motion
- DOF of the Joint = 3



Overview

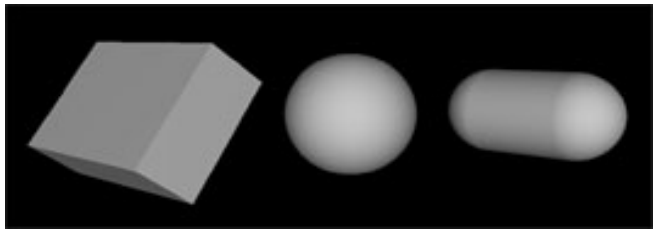
- Particle Dynamics
- Rigid Body Dynamics
- Collision and Contact
- Constraint/Joint
- **Introduction to VirtualPhysics**

VirtualPhysics

- C++ library for realtime multi-body dynamics simulation
- based on Lie group formulation
- arbitrary joint modeling
- collision detection between geometric primitives
- analytic or penalty based collision response
- contact response by solving LCP
- mixed dynamics: working with kinematically controlled objects
- collision like joint limit
- dynamic configuration: excessive force can break joints

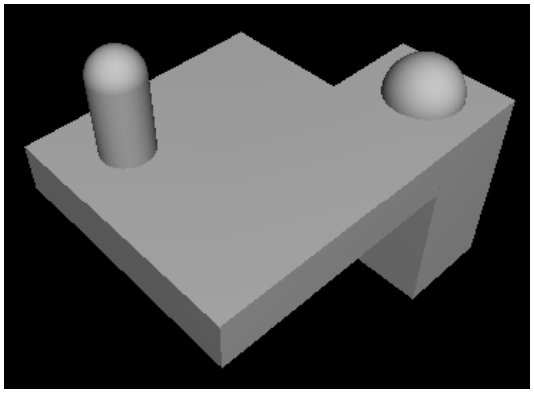
vpGeom

- Abstract class to represent primitive geometries



vpBox vpSphere vpCapsule

- Consider a rigid body as a set of primitive geometries



- Target of collision detection

vpBody

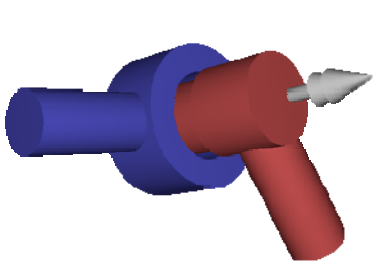
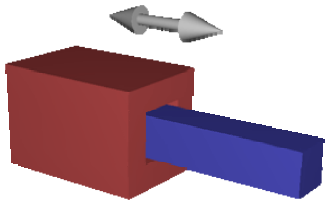
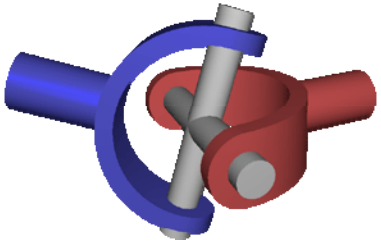
- A class to model rigid bodies
- consists of several vpGeom instances
- can be connected to another body with vpJoint.
- has a reference to vpMaterial
- can be set ground

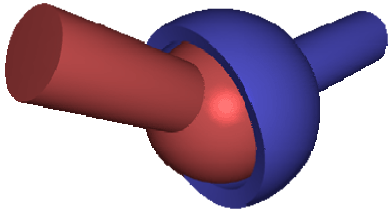
vpMaterial

- A class about material properties
  - density
  - coefficient of restitution
  - coefficient of static/dynamic/spinning friction

vpJoint

- Abstract class to connect two adjacent bodies
- Can be broken
  - , when a normal force/torque applied to the joint exceeds its threshhold.



vpRJointv **pPJoint** **vpUJoint** **vpSJoint** **vpBJoint**

vpWJoint **vpNDOFJoint**

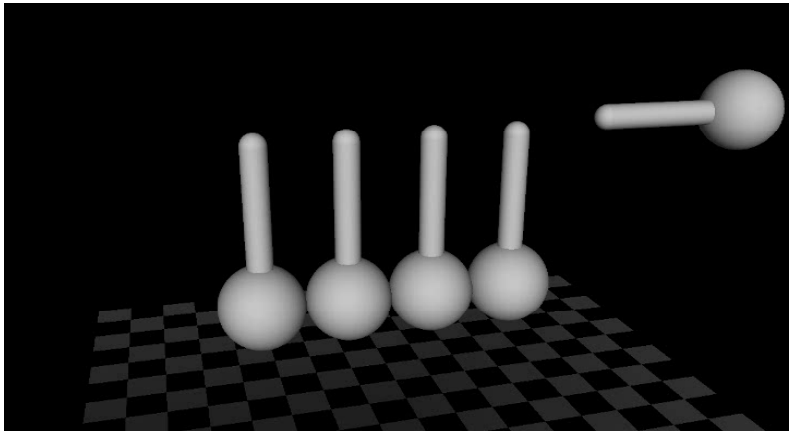
vpWorld

- A class to model a virtual world
- Add bodies to vpWorld
- Set global attributes such as gravity
- Set simulation attributes such as integration time step
- Initialize and run

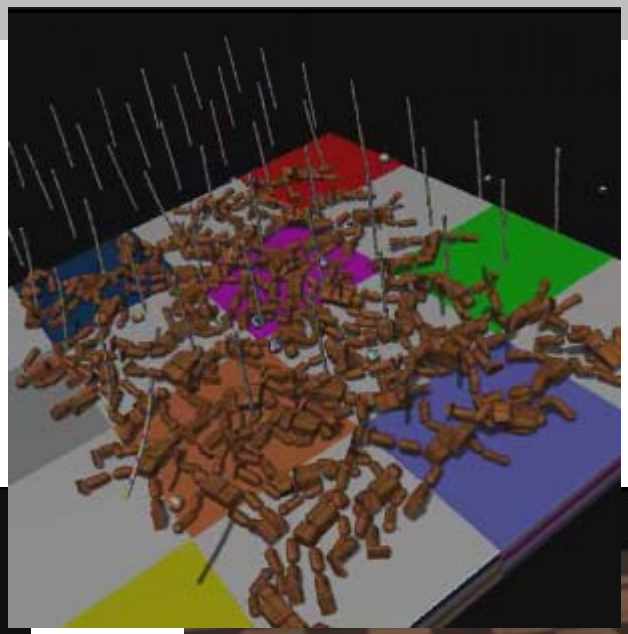
Example: a pedulum

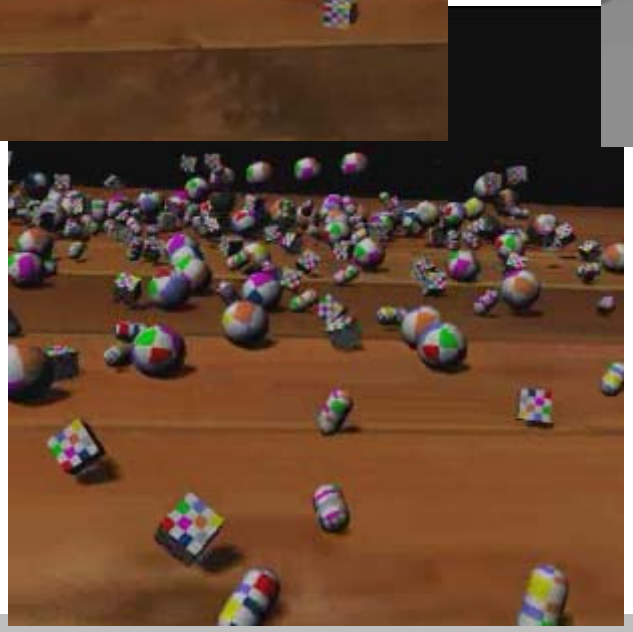
- Add two bodies and initialize a joint

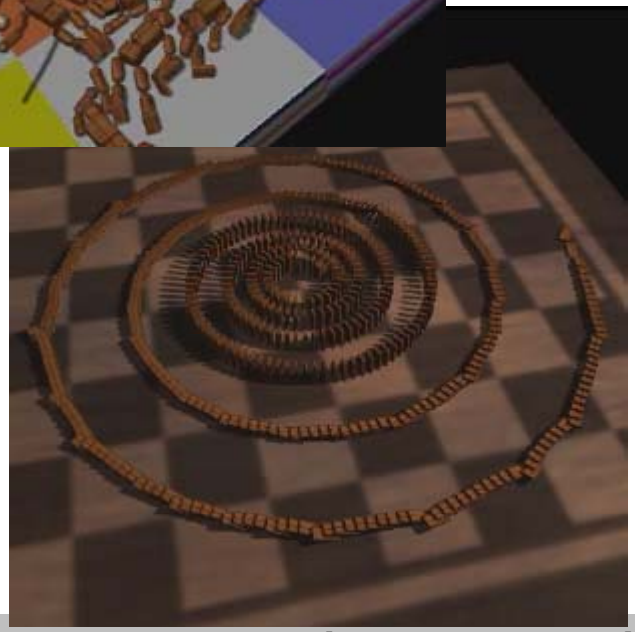
```
vpBody G, B;
vpSphere sph(0.5);
vpCapsule rod(0.2,2);
B.AddGeometry(&rod, Vec3(0));
B.AddGeometry(&sph, Vec3(0,0,-1));
vpRJoint J;
G.SetJoint(&J, Vec3(0));
B.SetJoint(&J, Vec3(0,0,1));
J.SetAxis(Vec3(0,1,0));
G.SetGround();
vpWorld world;
world.AddBody(&G);
world.Initialize();
while ( true )
    world.StepAhead();
```



http://virtualphysics.imrc.kist.re.kr







Question?

Feel free to email me

jwkim@imrc.kist.re.kr

Collision Detection for Real-time Physics

Young J. Kim

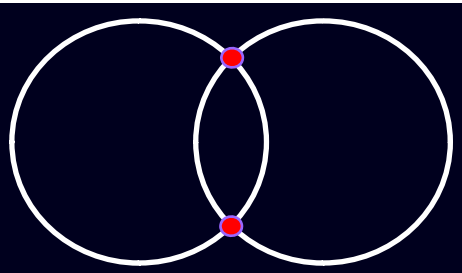
http://graphics.ewha.ac.kr

Dept of Computer Sci and Eng

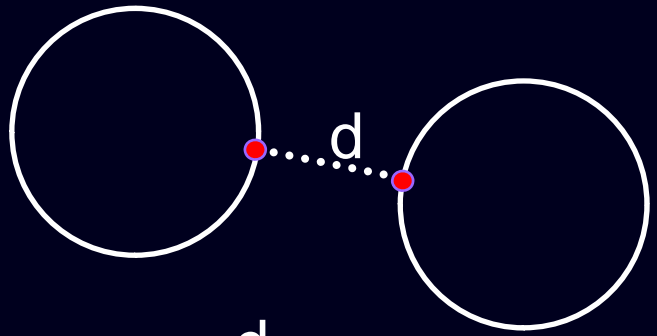
Ewha Womans University

Definitions

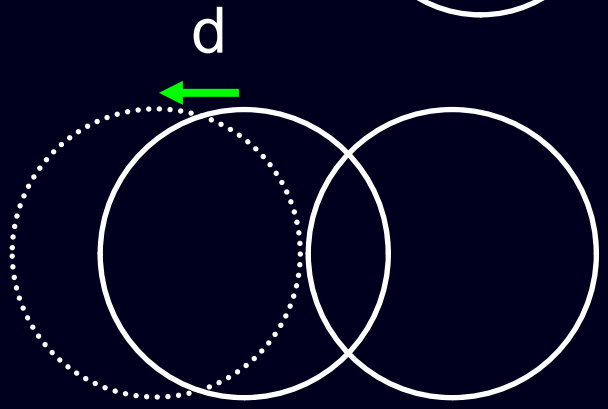
- **Collision detection** :
  check whether two objects overlap in space



- **Separation distance** :
  minimum Euclidean distance



- **Penetration depth**:
  minimum Euclidean distance to separate intersecting objects



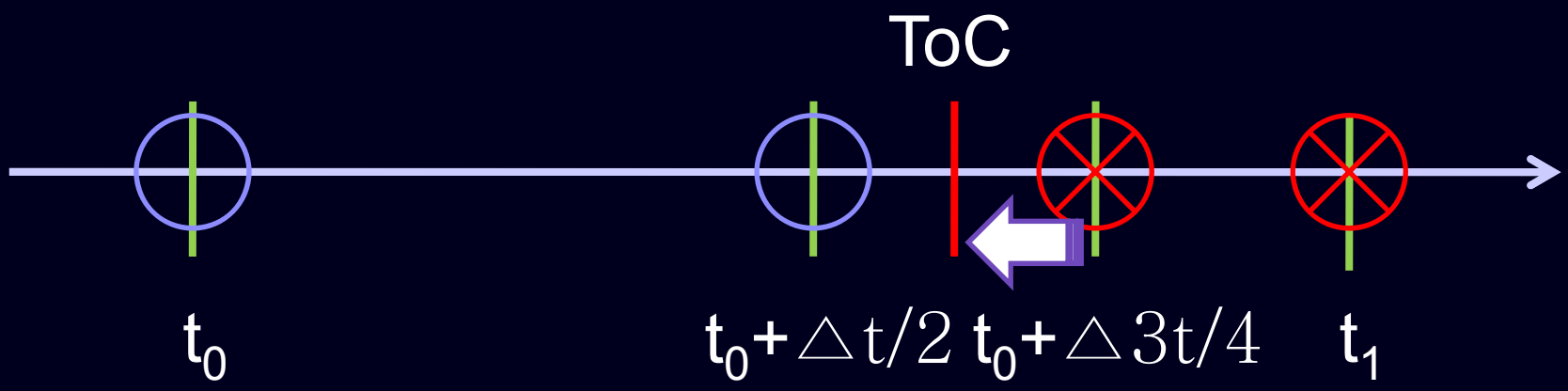
Time of Contact Computation

- Methods
  1. Bisection using collision detection and separation distance
  2. Penalty-based using penetration depth
  3. Continuous collision detection

Bisection Search

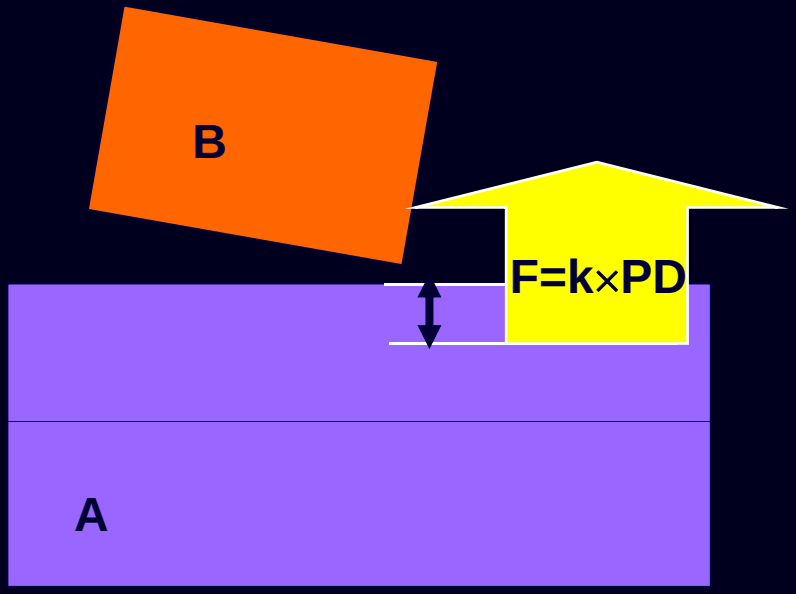
While {collided or distance(A,B)>ε}, do

bisect $[t_0, t_1]$;



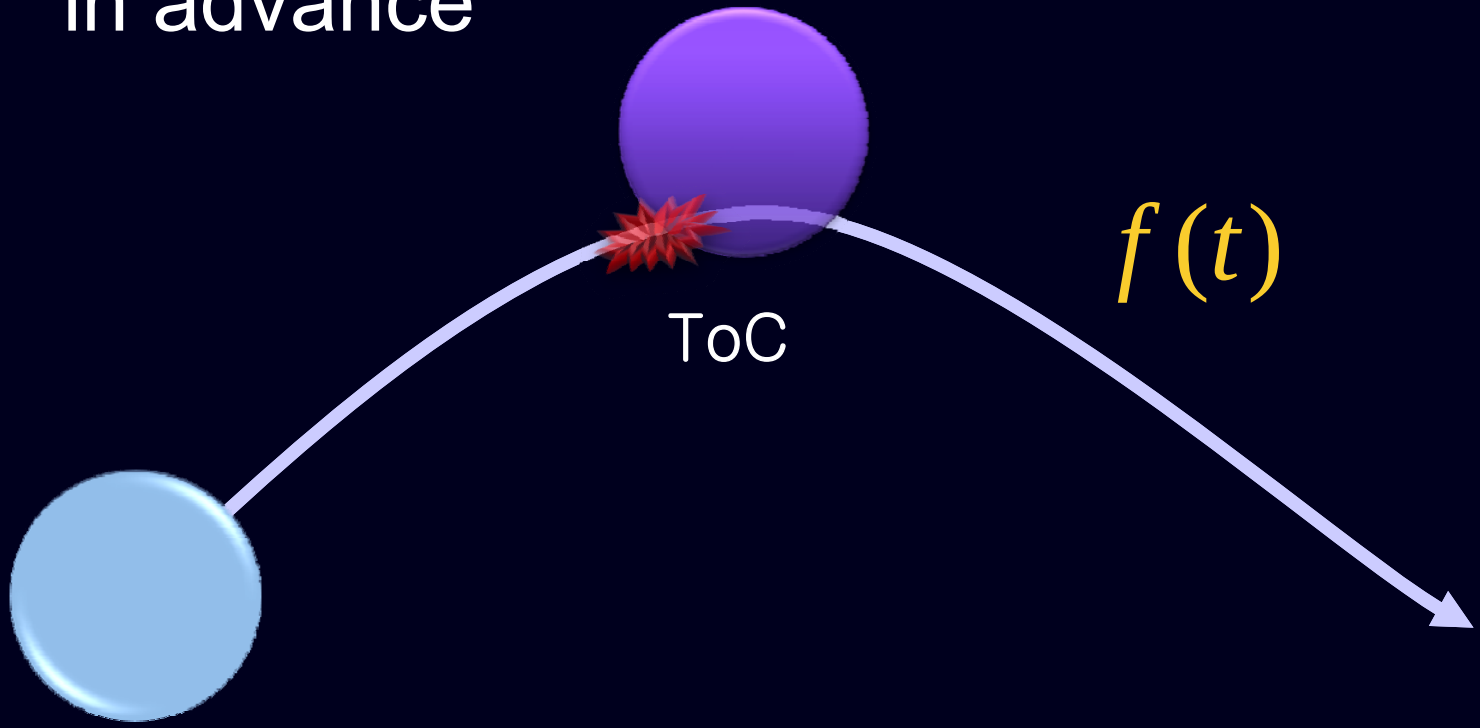
Penalty-based Method

- Penetration depth (PD) is required for computing penalty-based contact response



Continuous Collision Detection

- Motion trajectory ***f(t)*** is known/computed in advance



Contents

- Separation distance
- Collision detection
- Continuous collision detection
- Penetration depth computation

Separation Distance and Collision Detection

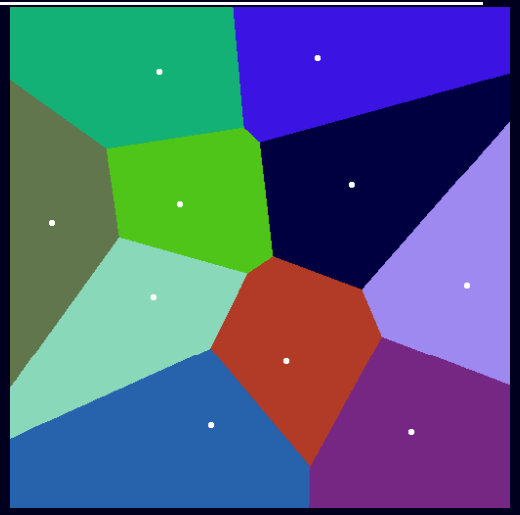
- Voronoi Marching [EL00,EL01]

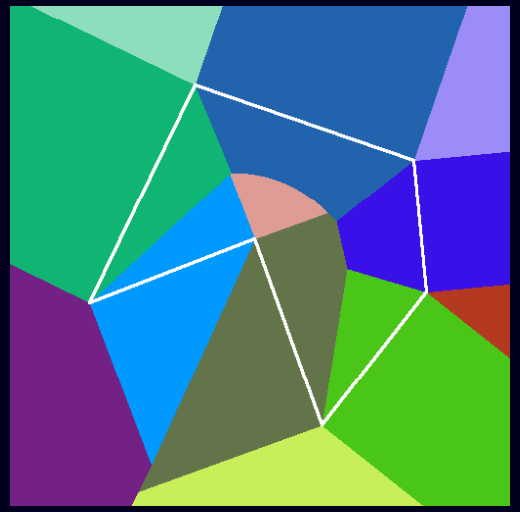
Voronoi Marching Algorithm

- Voronoi Diagram (VD)

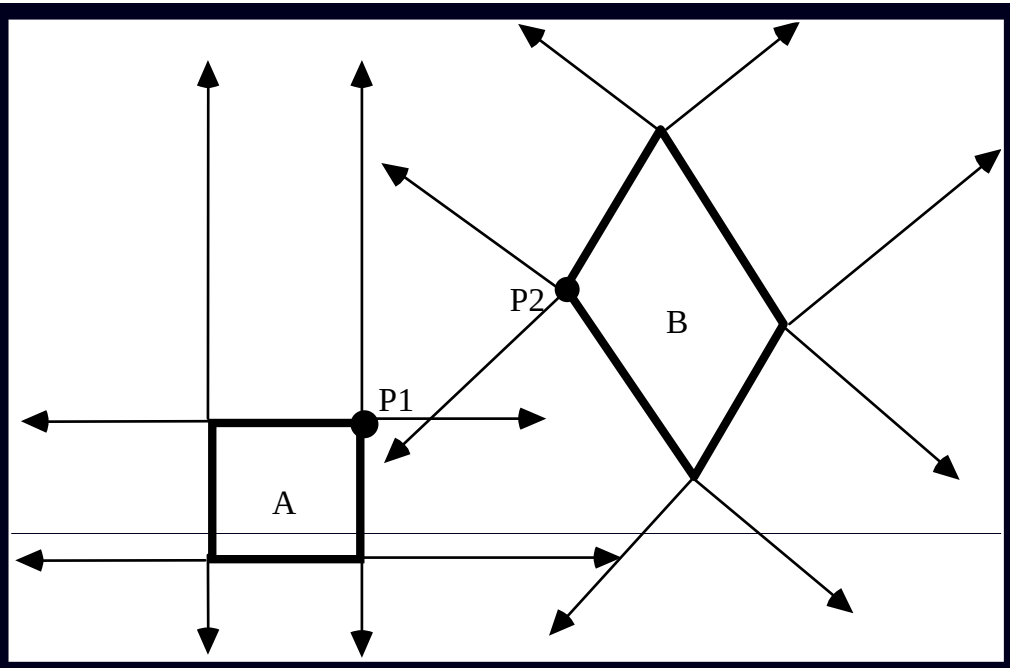
- Generalized VD: The extension of the Voronoi diagram to higher dimensional features, e.g. V,E,F of a polyhedron.

- FACTS:
  - In general, the GVD has quadratic surface boundaries
  - If the polyhedron is convex, then its GVD has planar boundaries.





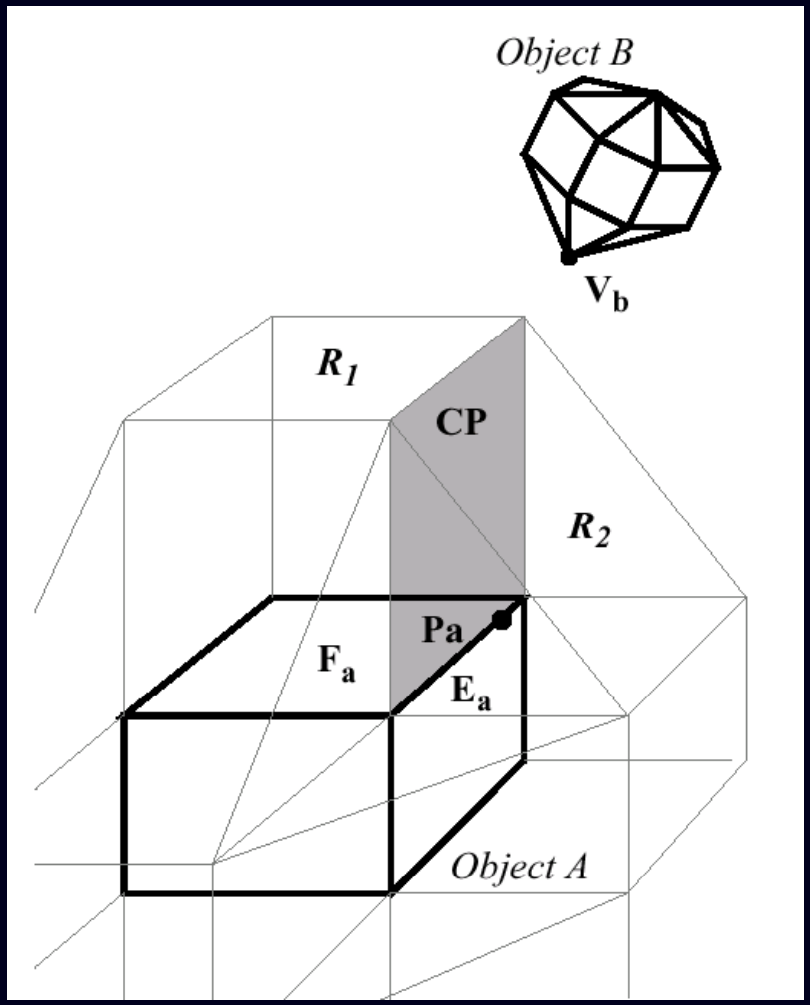
Simple 2D Example



$P_2 \subset Vor(P_1)$
$P_1 \subset Vor(P_2)$

P1 and P2 are the pair of closest points between A and B. Note P1 and P2 lie within the Voronoi regions of each other

Voronoi Marching



$$V_b \not\subset Vor(F_a)$$

$$\downarrow$$

$$V_b \subset Vor(E_a)$$

$$\downarrow$$

$$P_a \subset Vor(V_b)$$

$$\downarrow$$

$$\vdots$$

Collision Detection

- Original VM algorithm falls into an infinite loop if there is penetration

- Improved versions
  - V-Clip [Mir98]
    - More robust and checks for CD more easily
  - SWIFT [EL00]
    - Multi-resolution
  - SWIFT++ [EL01]
    - Can handle non-convex polyhedra

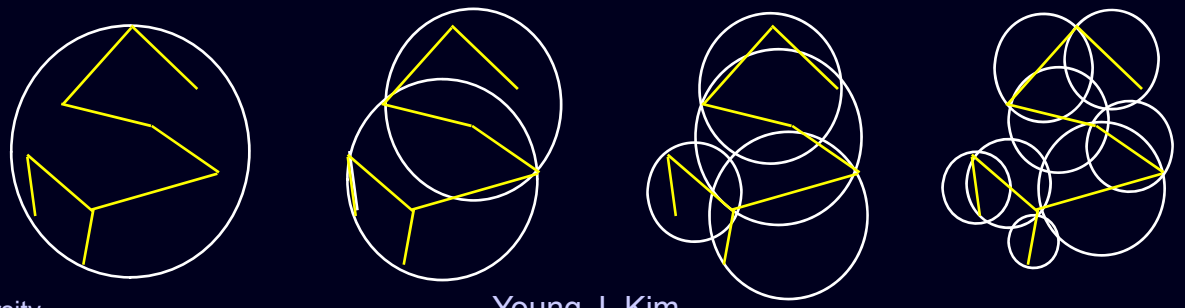
Collision Detection and Separation Distance Query

General Polygonal Models

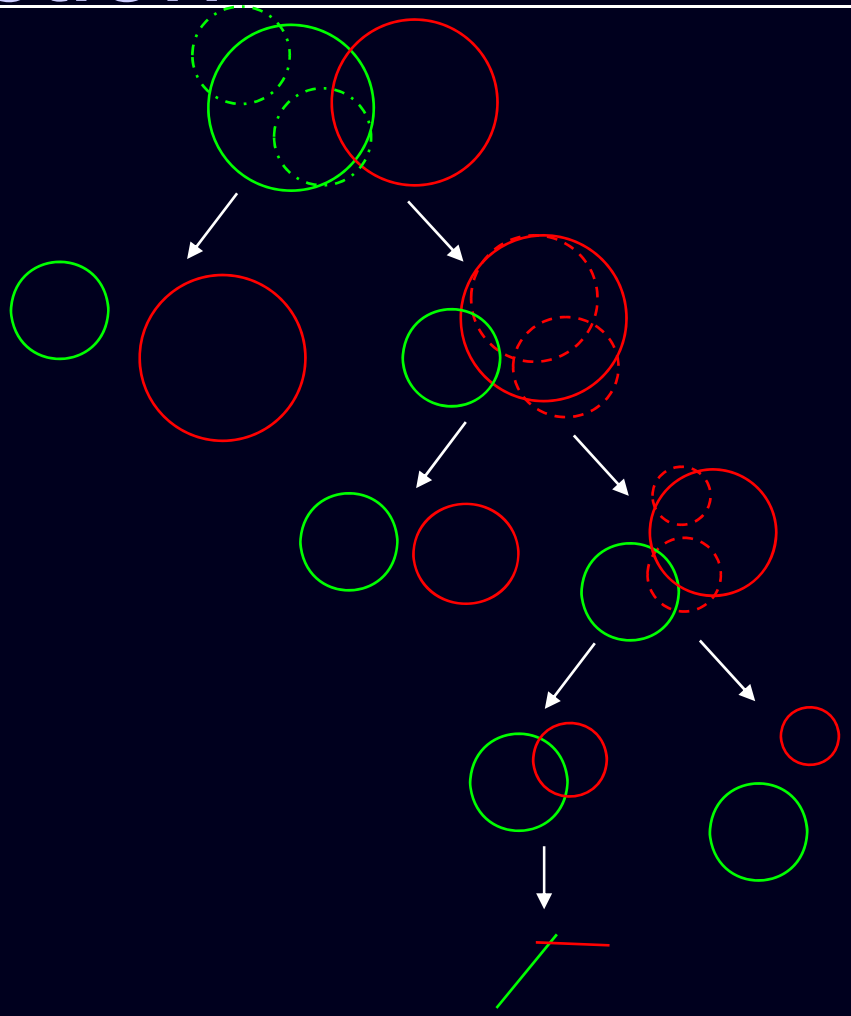
- Bounding Volume Hierarchies (BVHs)
- OBB [GLM96]

Bounding Volume Hierarchies

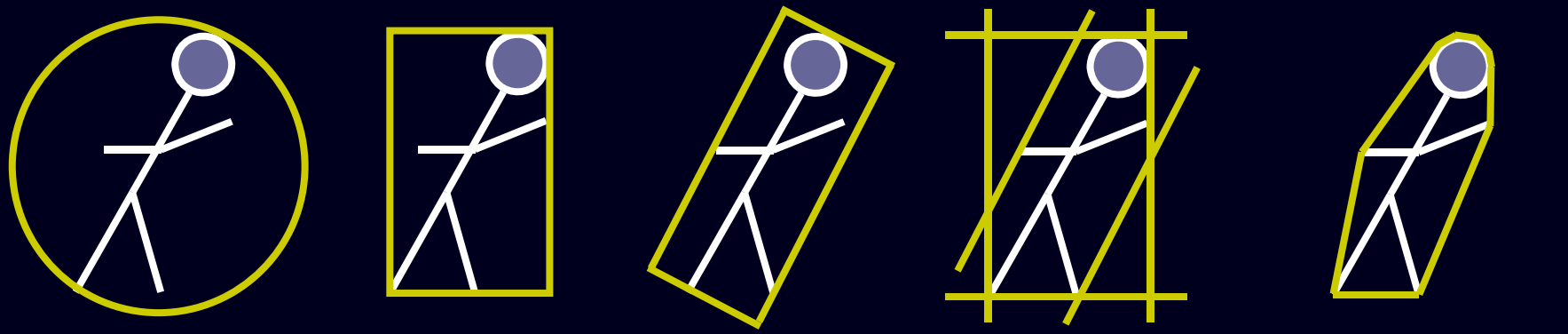
- Model Hierarchy:
  - Each node has a simple volume that bounds a set of triangles
  - Children contain volumes that each bound a different portion of the parent's triangles
  - The leaves of the hierarchy usually contain individual triangles
- A binary BVH



BVH-Based Collision Detection



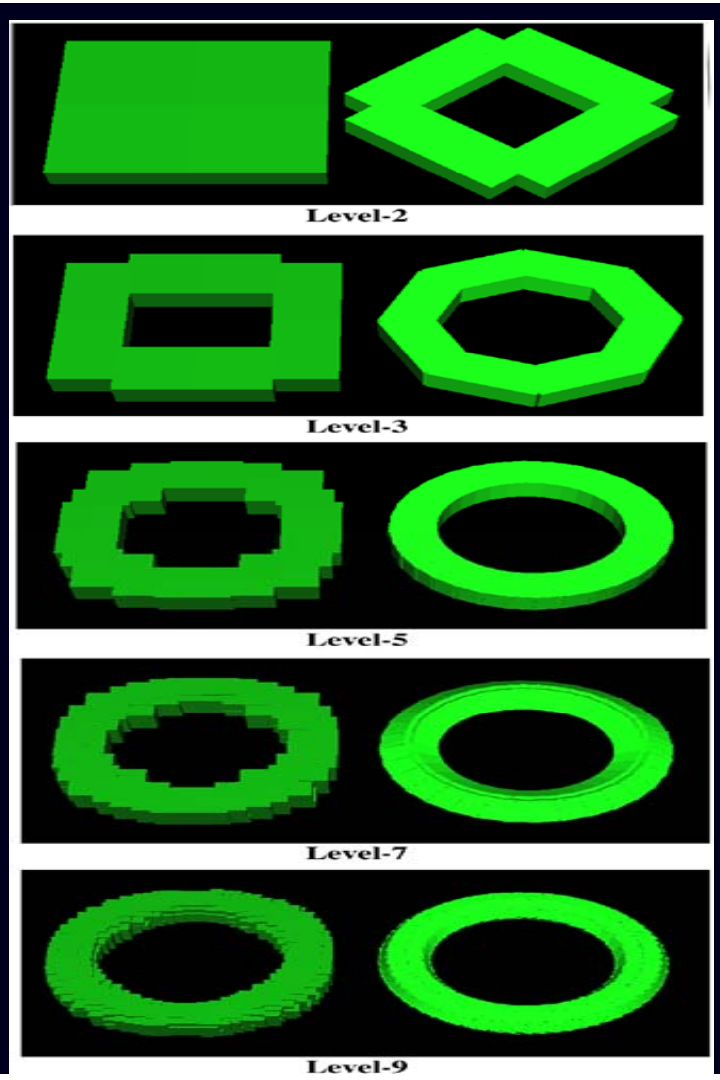
Trade-off in Choosing BV's



Increasing complexity & tightness of fit

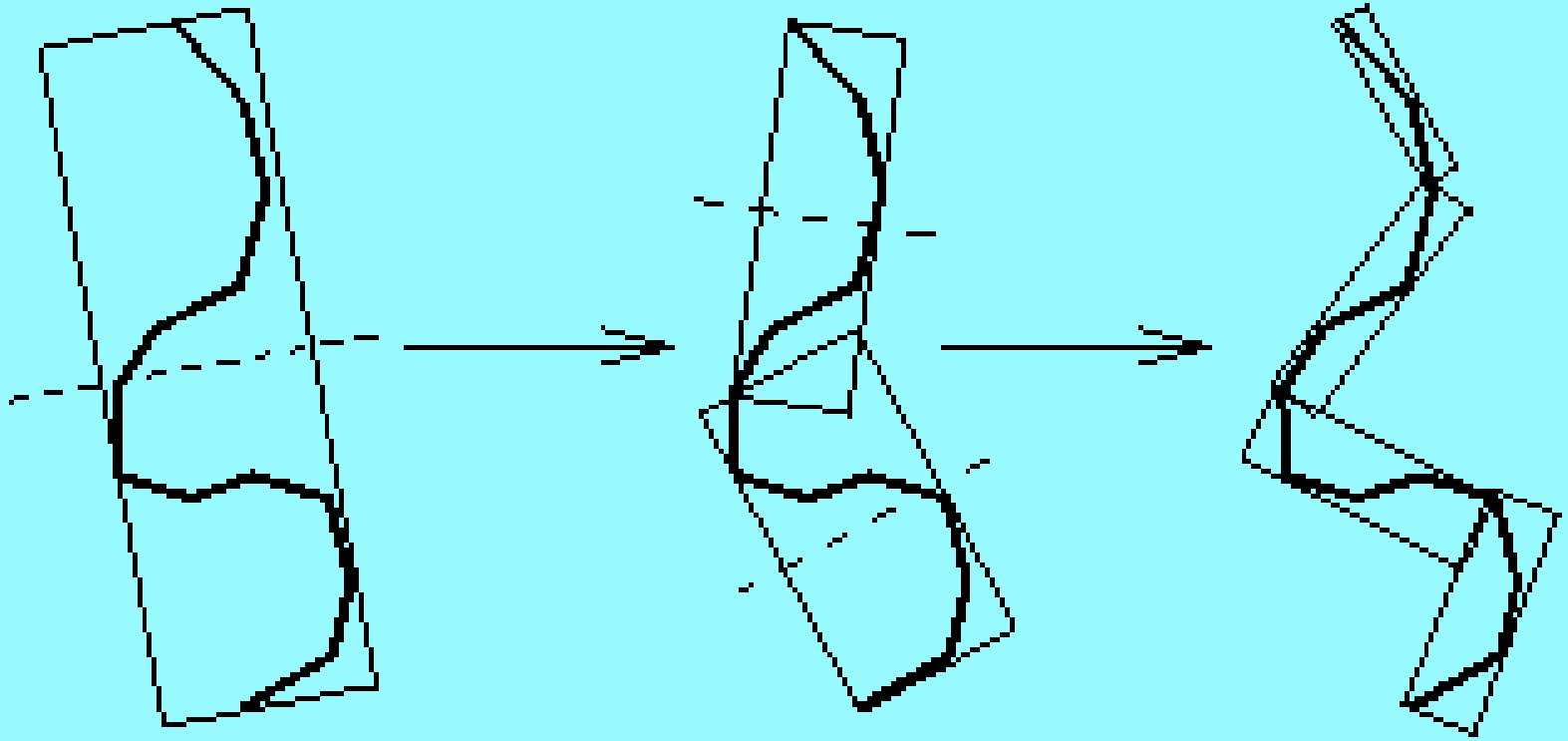
Decreasing cost of (overlap tests + BV update)

Example: AABB's Vs. OBB's



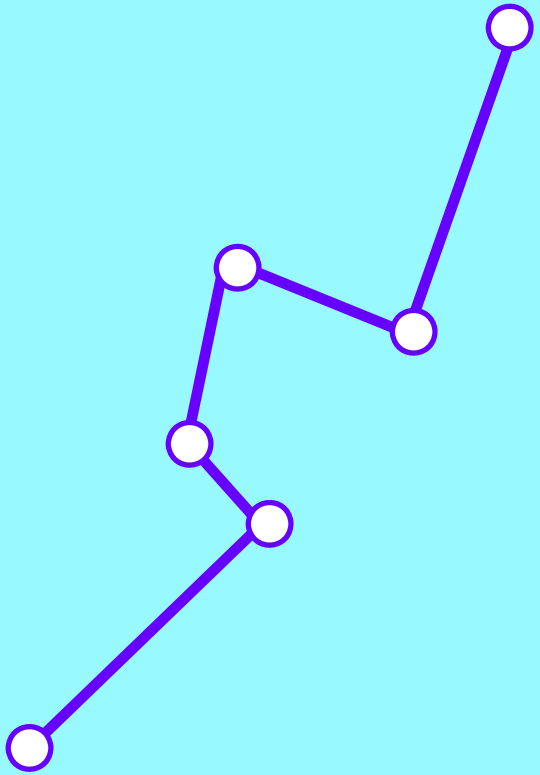
Approximation of a Torus

Building an OBBTree



Recursive top-down construction: partition and refit

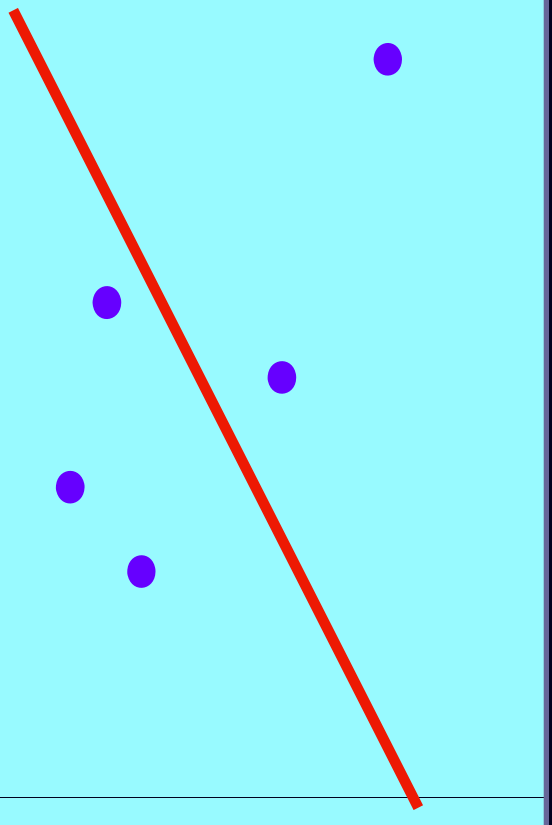
Building an OBB Tree



Given some polygons, consider their vertices...

Building an OBB Tree

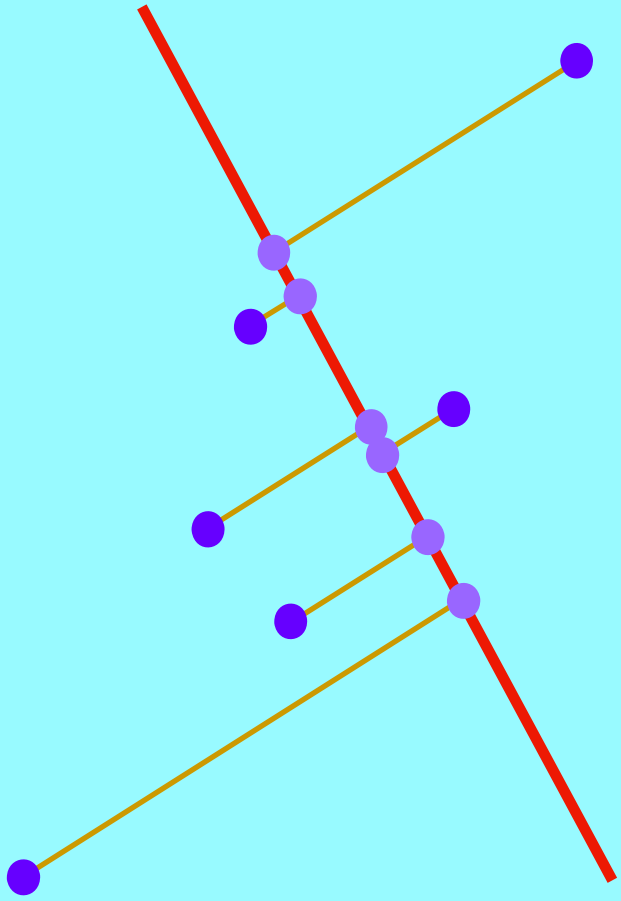
… and an arbitrary line



Building an OBB Tree

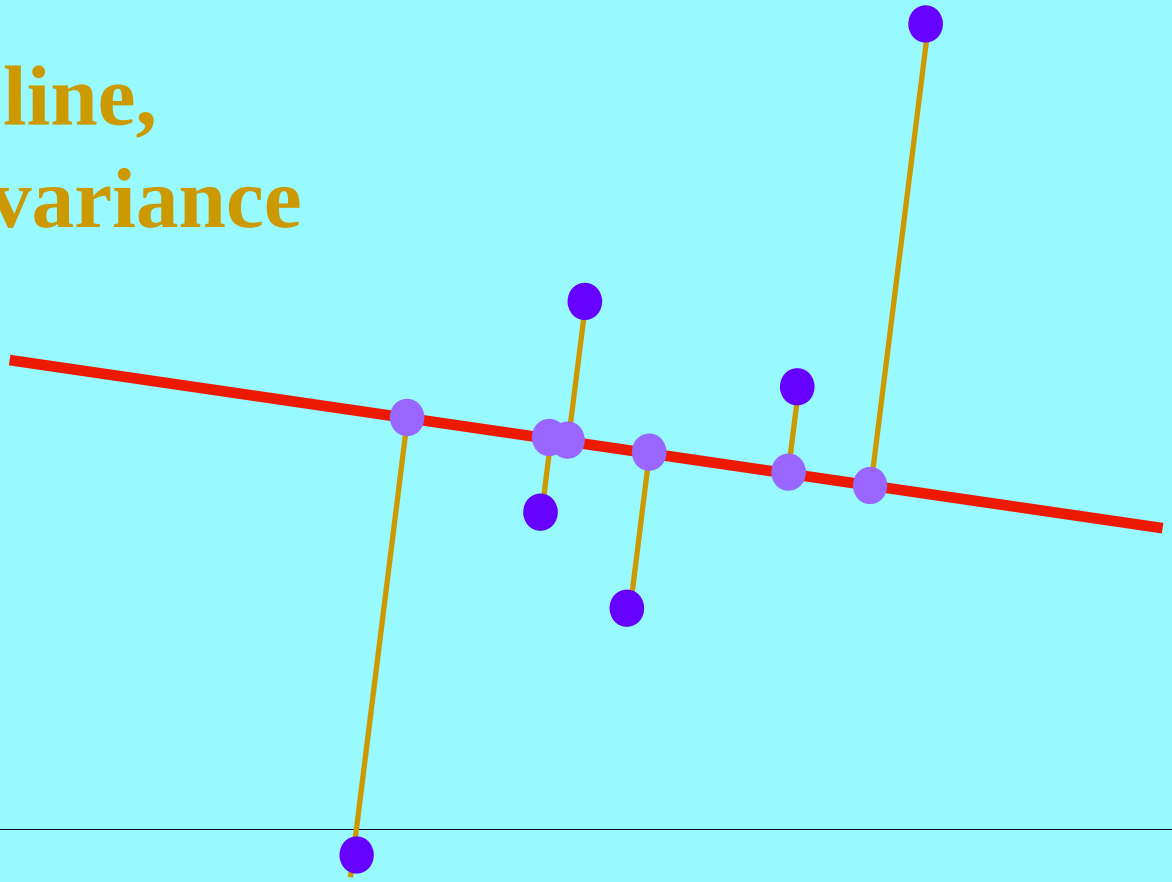
Project onto the line

Consider variance of distribution on the line



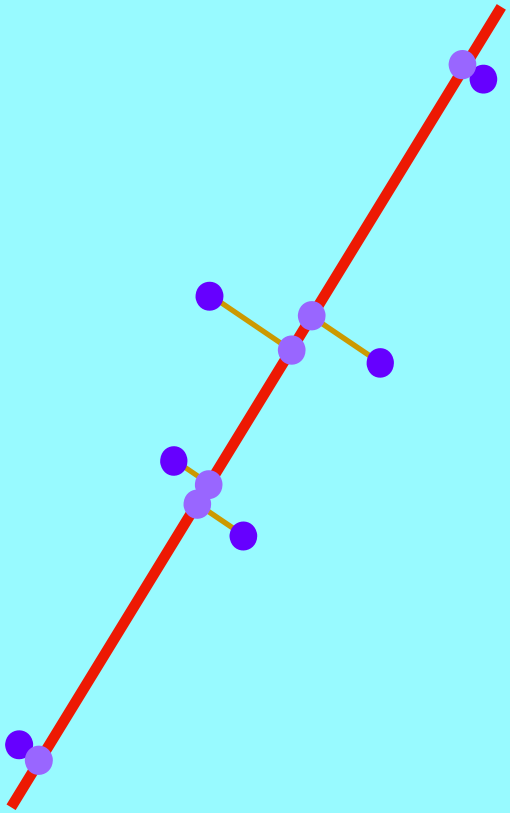
Building an OBB Tree

Different line,
different variance

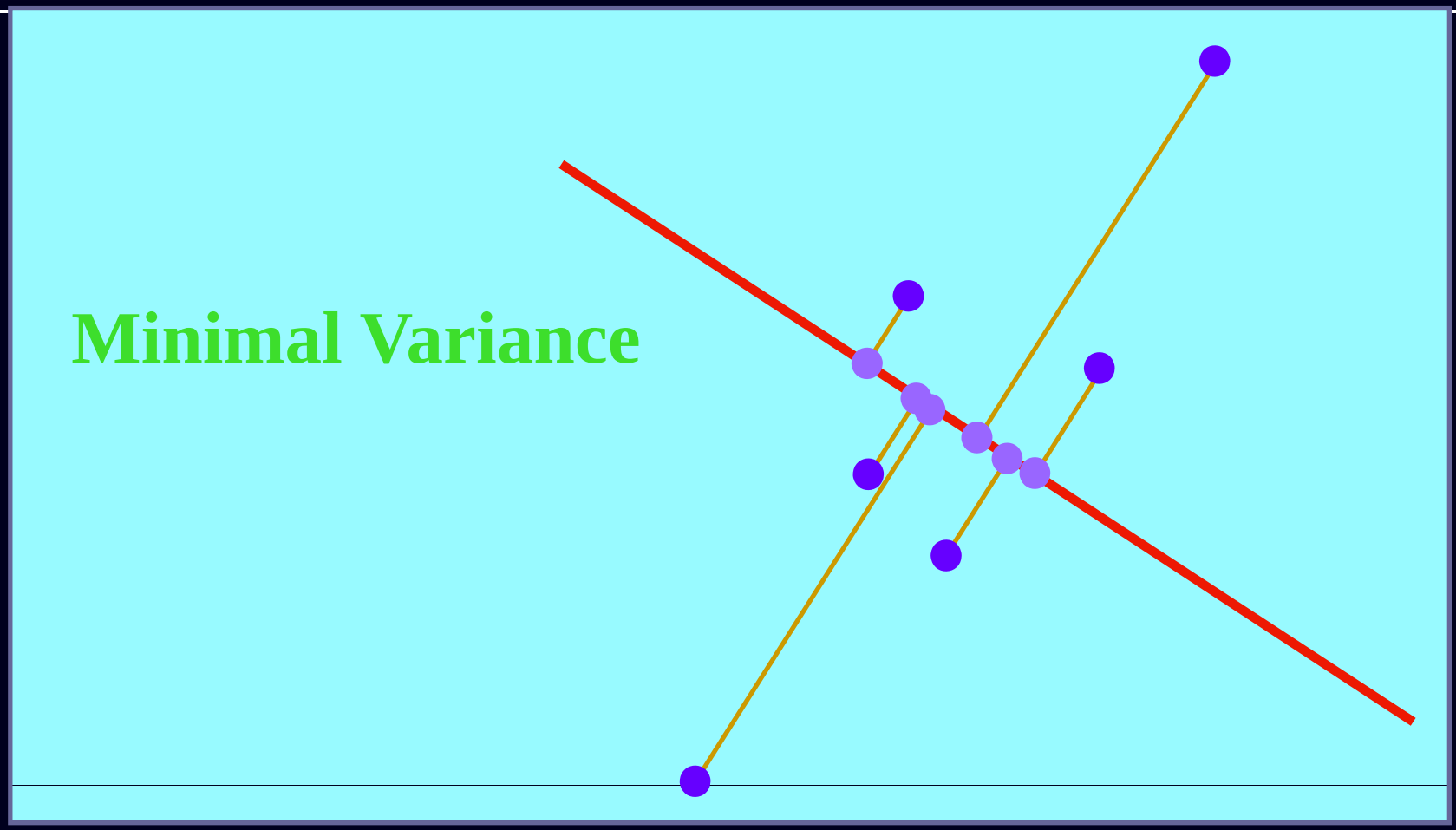


Building an OBB Tree

Maximum Variance

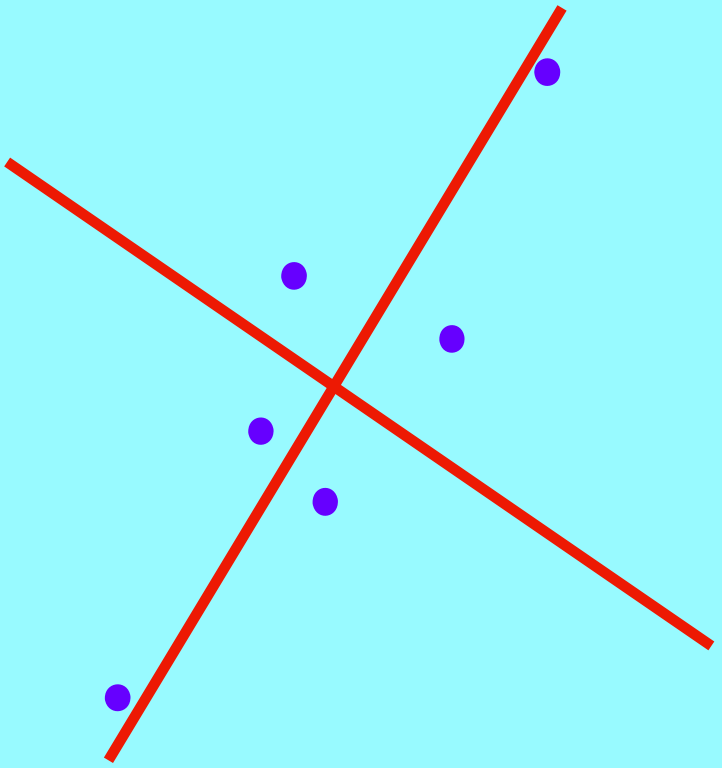


Building an OBB Tree



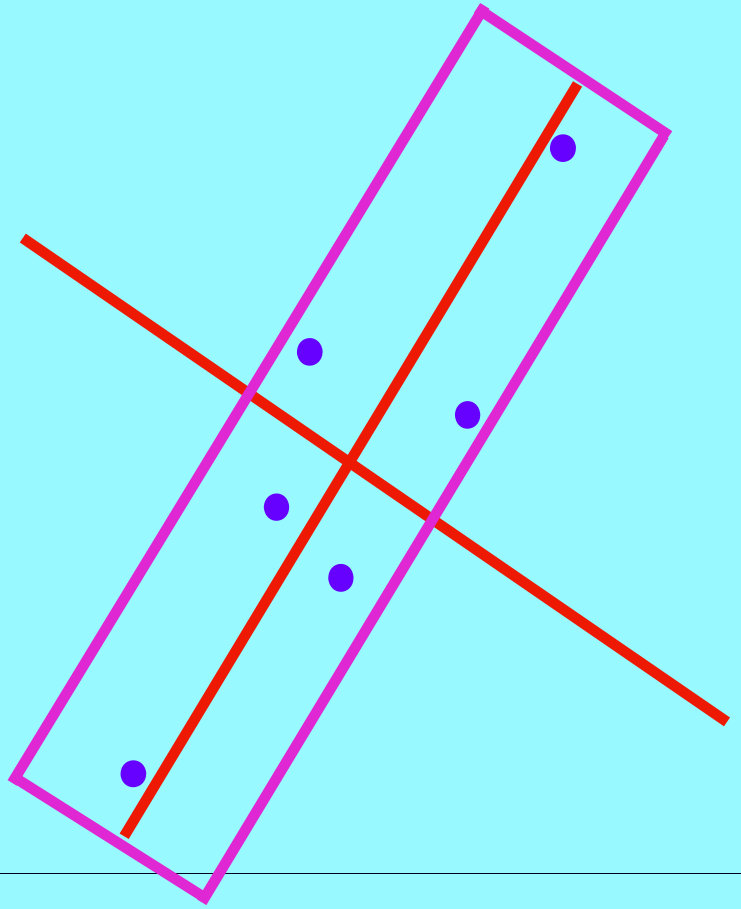
Building an OBB Tree

Given by eigenvectors
of covariance matrix
of coordinates
of original points



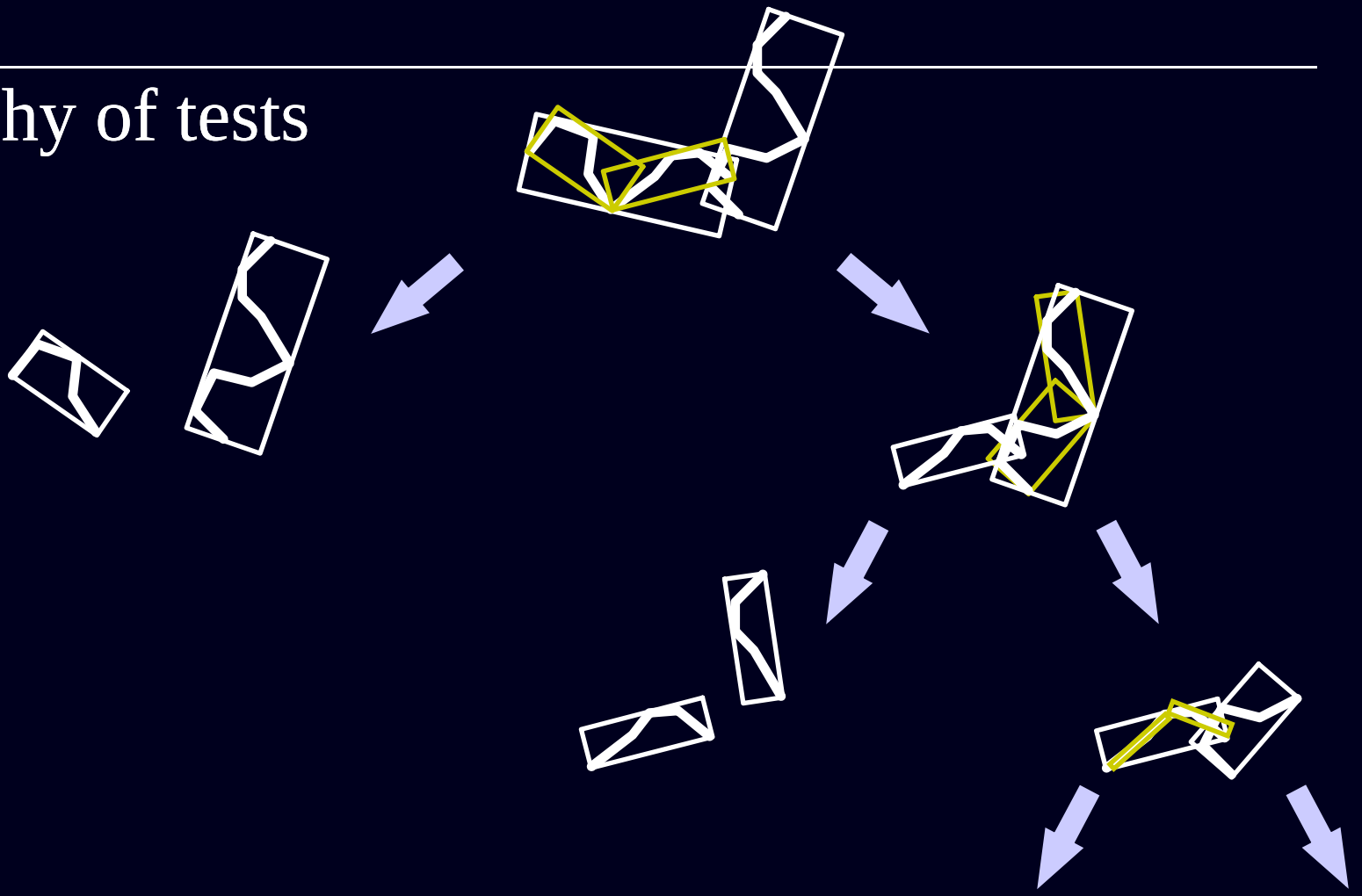
Building an OBB Tree

Choose bounding box oriented this way



Tree Traversal

Hierarchy of tests



Separating Axis Theorem

Two polytopes A and B are disjoint iff there exists a separating axis which is:

perpendicular to a face from either

or

perpendicular to an edge from each

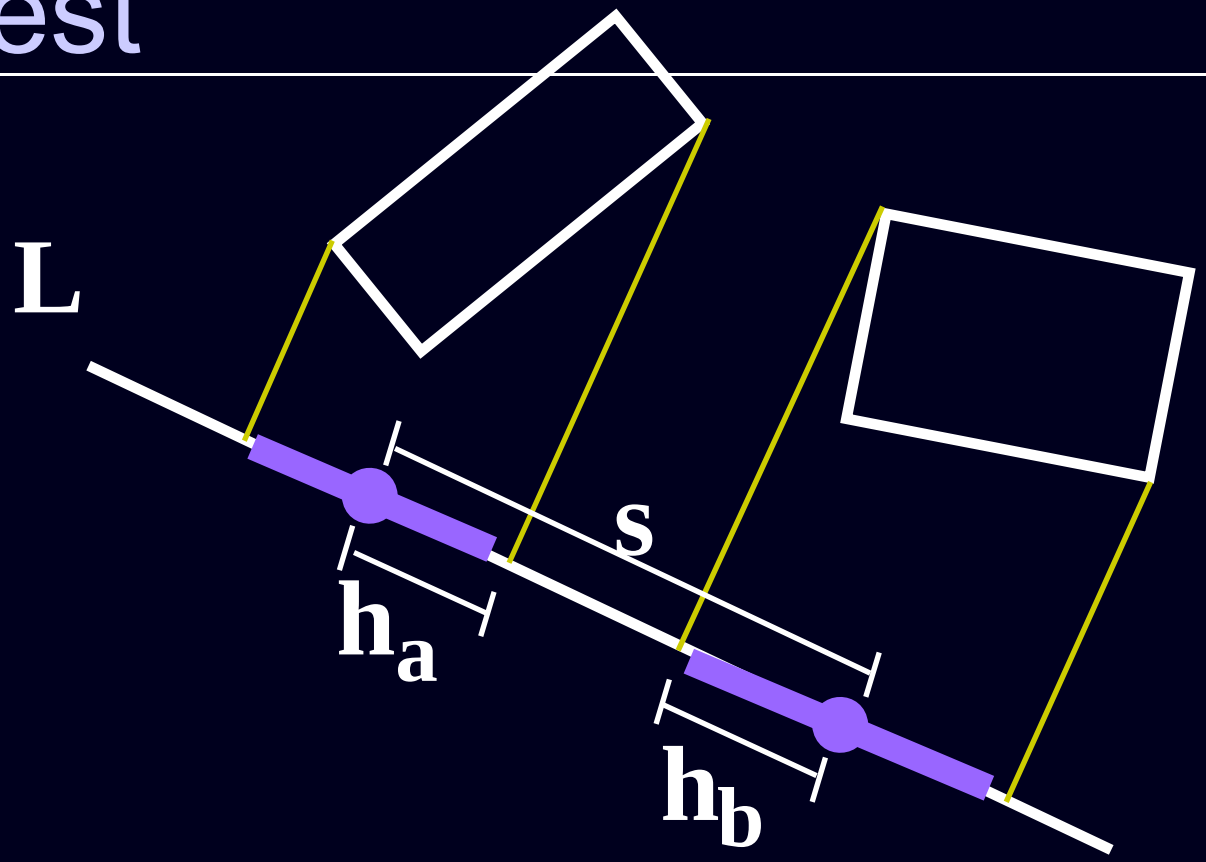
Implications of Theorem

Given two generic polytopes, each with E edges and F faces, number of candidate axes to test is:

$$2F + E^2$$

OBBs have only E = 3 distinct edge directions, and only F = 3 distinct face normals. OBBs need at most 15 axis tests.

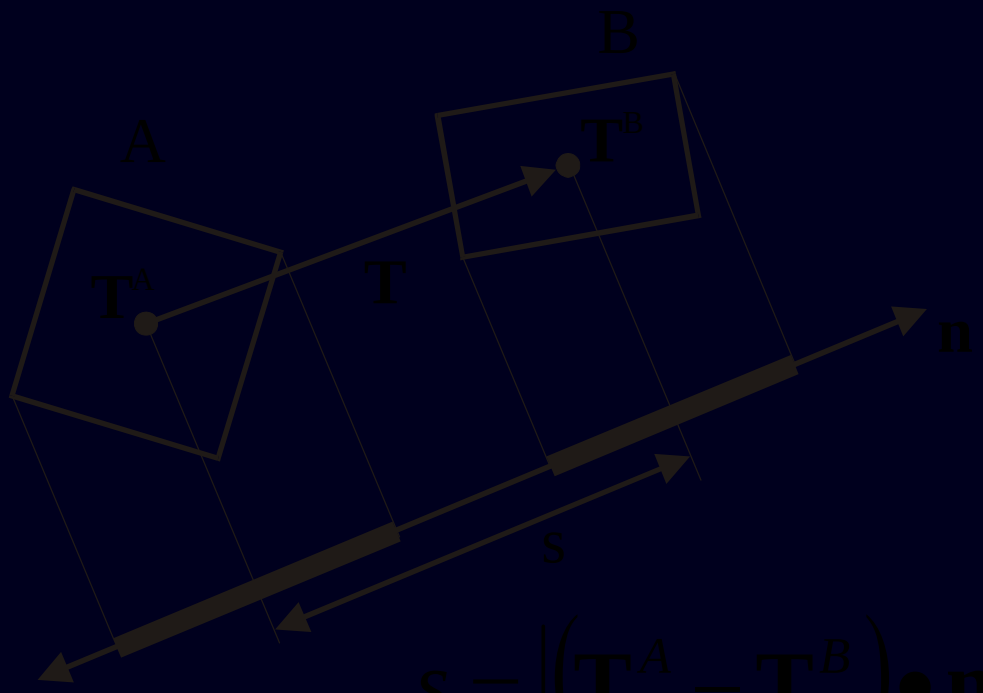
OBB Overlap Test: An Axis Test



L is a separating axis iff: $s > h_a + h_b$

OBB Overlap Test: Axis Test Details

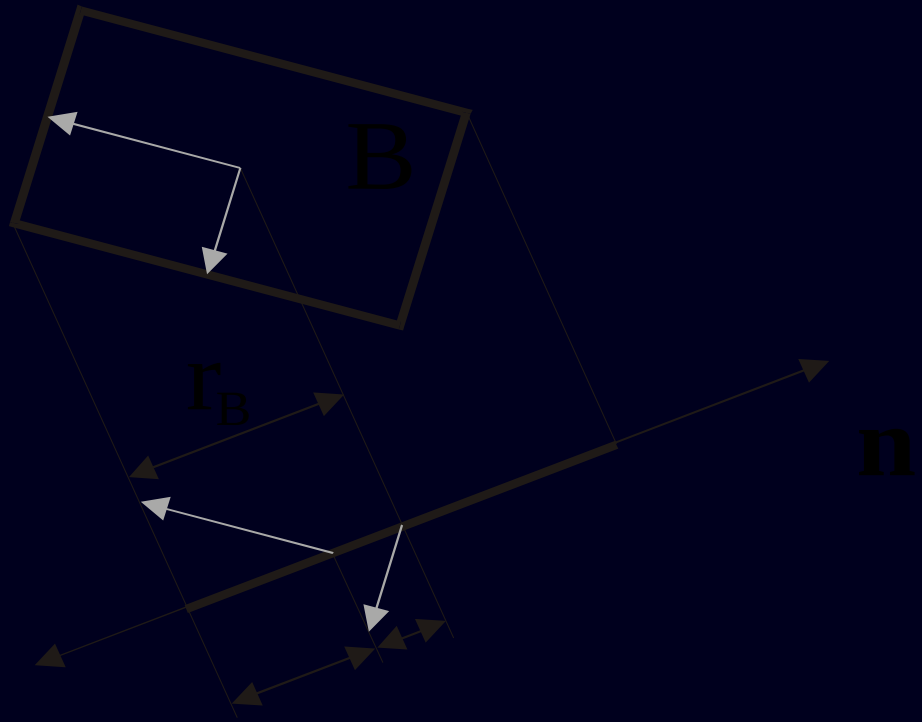
Box centers project to interval midpoints, so midpoint separation is length of vector **T**'s image.



$$s = \left| \left(\mathbf{T}^A - \mathbf{T}^B \right) \bullet \mathbf{n} \right|$$

OBB Overlap Test: Axis Test Details

- Half-length of interval is sum of box axis images.



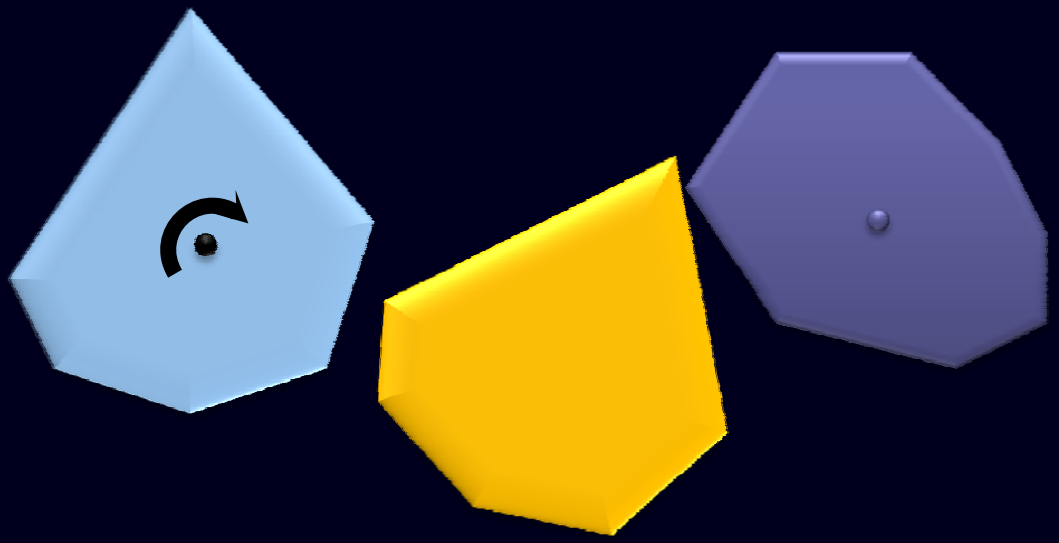
$$r_B = b_1\left|\mathbf{R}_1^B \cdot \mathbf{n}\right| + b_2\left|\mathbf{R}_2^B \cdot \mathbf{n}\right| + b_3\left|\mathbf{R}_3^B \cdot \mathbf{n}\right|$$

Continuous Collision Detection

- Non-convex Polyhedral Model [ZLK06]

Conservative Advancement

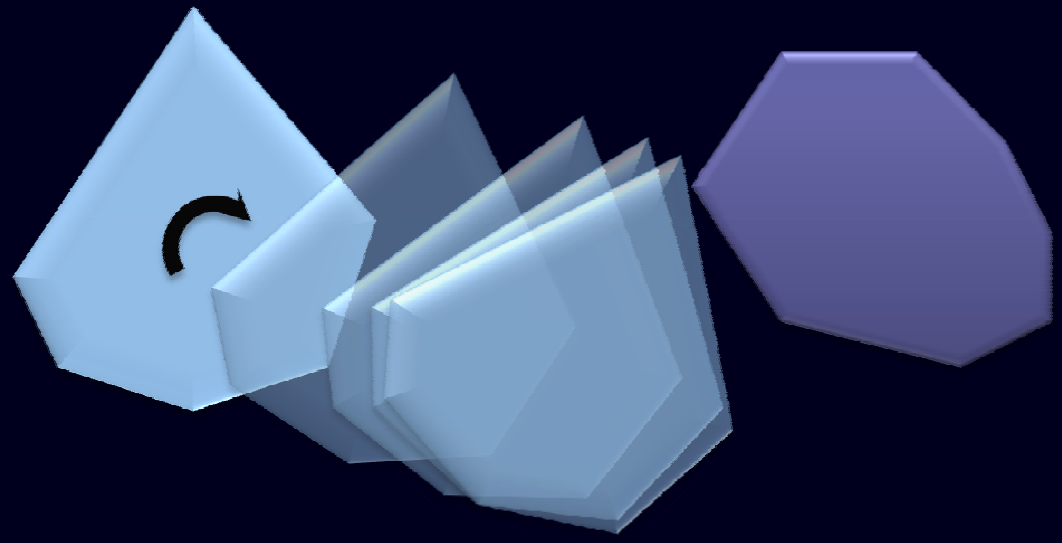
- Goal
  - Assume objects are *convex* [Mirtich 96, 00]
  - Find the 1st time of contact (ToC) of a moving object



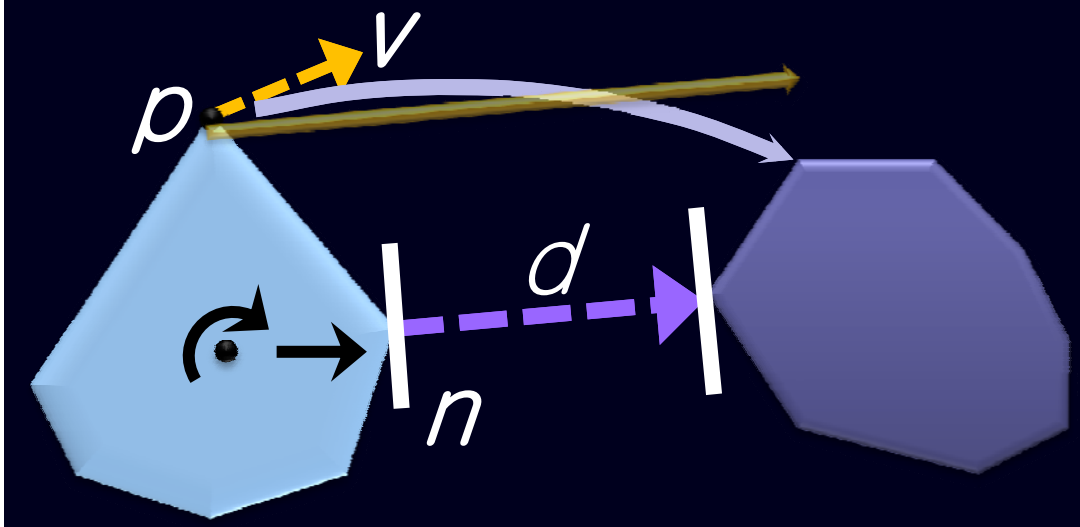
Conservative Advancement

1. Find a step size Δt_i to conservatively advance the object without collision
2. Repeat until inter-distance < ε

$$ToC = \Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4$$



Calculating Δt in CA



d, n: closest distance direction vector

v: velocity

$$\int_0^{\Delta t} \left| \nabla \left(\int_0^{\Delta t} \max(|v(t)|) dt \right) \cdot n(t) \right| dt = \mu$$

Calculating Δt in CA

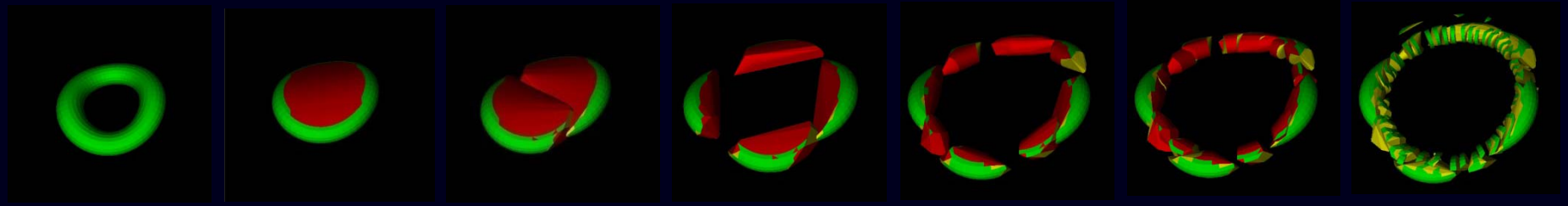
$$\mu \Delta t \le d \qquad \therefore \Delta t \le \frac{d}{\mu}$$

$$\int_0^{\Delta t} |v(t) \bullet n(t)| \, dt \le \int_0^{\Delta t} \max(|v(t) \bullet n(t)|) \, dt \le d$$

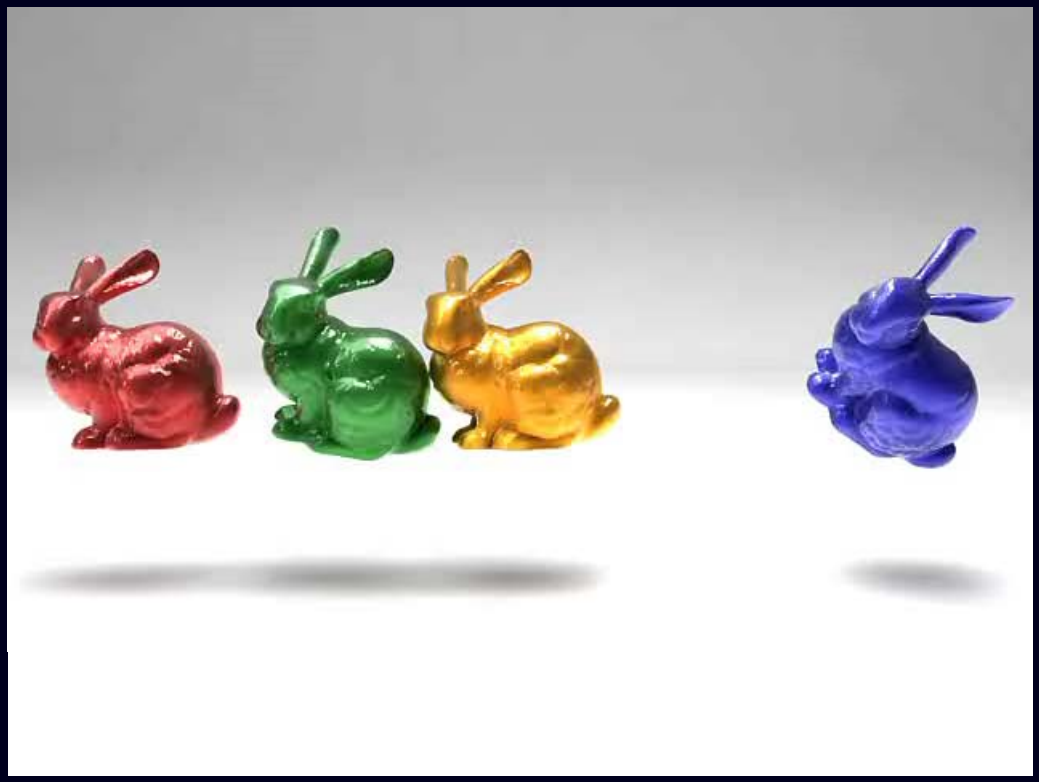
$= \mu$

Extension to Non-Convex Models

- Build a hierarchy of decomposed convex pieces and perform CA *hierarchically*
- Source codes : http://graphics.ewha.ac.kr/FAST



Bunny vs. Bunny



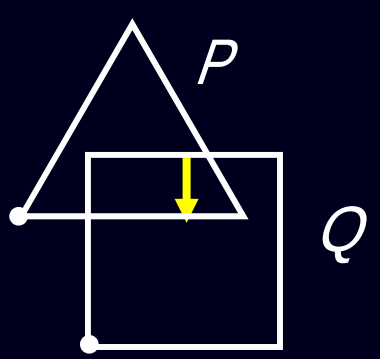
70K triangles/bunny 110 FPS # of iterations 4.7

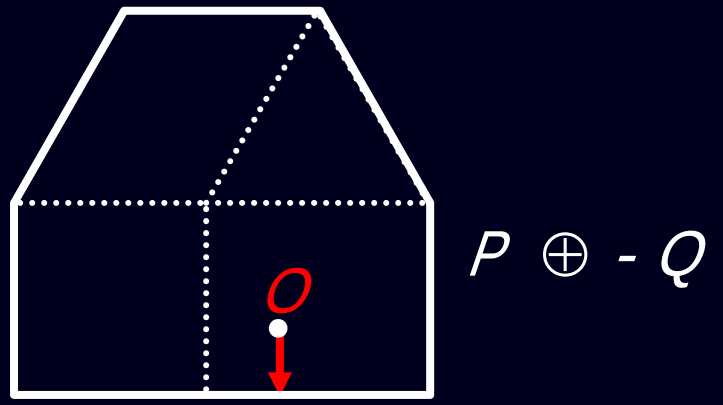
Penetration Depth Computation

- Convex Polytopes [KLM04]

DEEP Algorithm

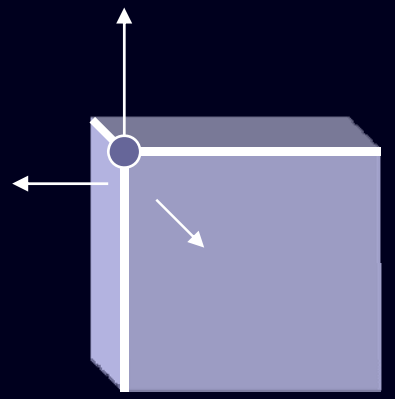
- Minkowski Sum
  - $P \oplus Q = \{ p+q \mid p \in P, q \in Q \}$
  - $P \oplus -Q = \{ p-q \mid p \in P, q \in Q \}$, a.k.a. CSO or TCSO
- PD := minimum distance btwn O and $\partial(P + -Q)$



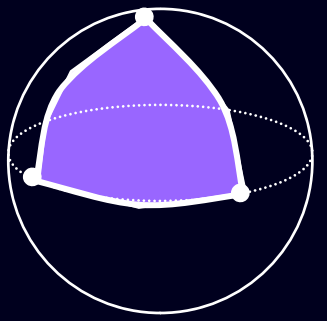


Gauss Map

- Gauss Map ($F \rightarrow S^2$) : Dual mapping from feature space to normal space
  - Face $\rightarrow$ point (outward normal)
  - Edge $\rightarrow$ great arc (locus of normals of two adjacent faces)
  - Vertex $\rightarrow$ region (bounded by *arc*s)

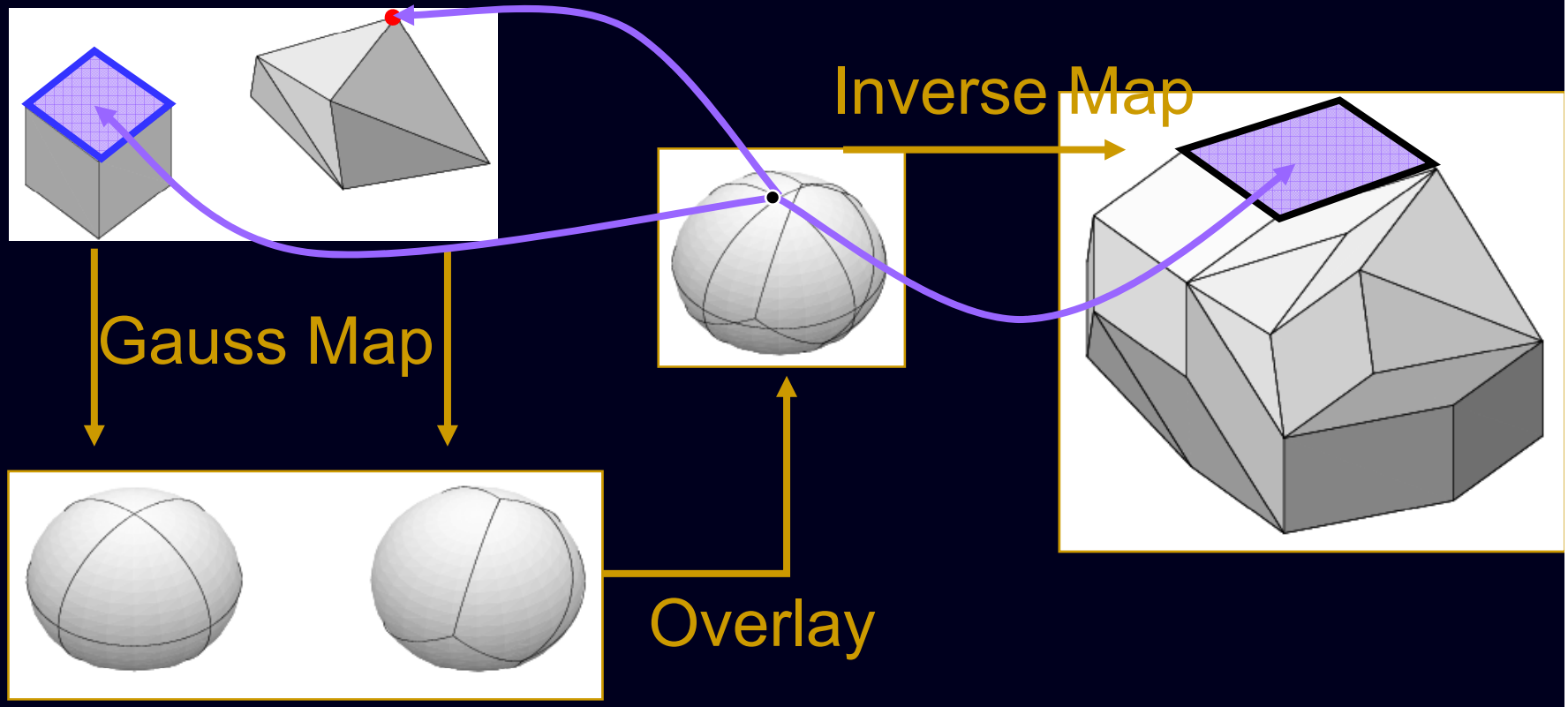


F



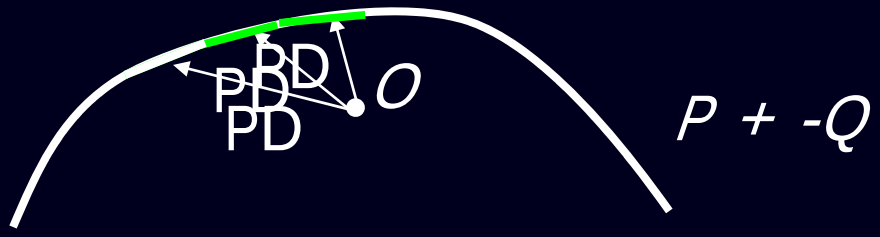
S^2

Minkowski Sum



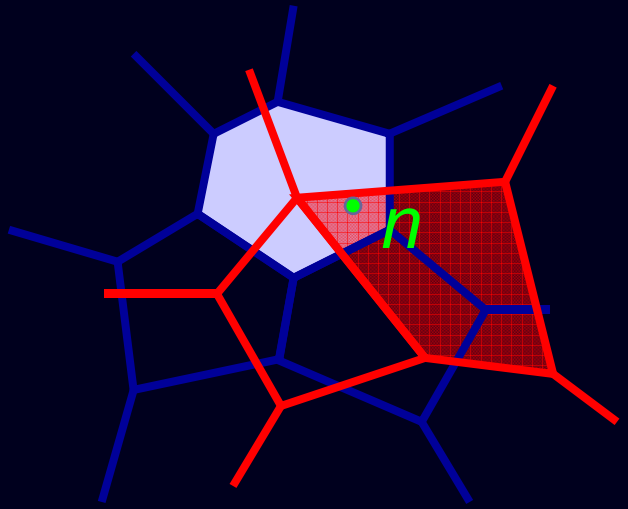
Overview

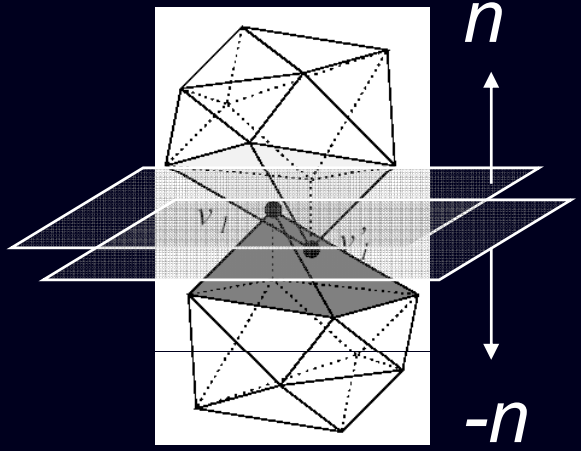
- Localized computations for Gauss map and overlay
- Iterative optimization
  - Identify an initial feature for *walk*
  - Measure the current PD
  - March toward the local optimum



Initialization

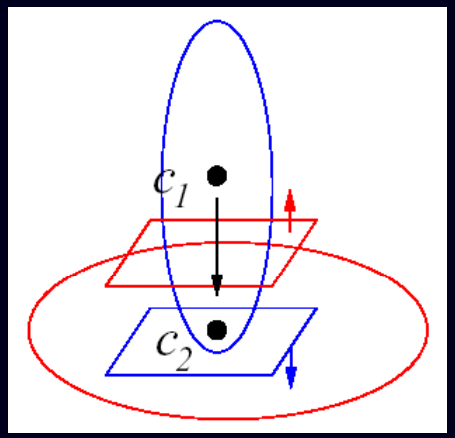
- Find a subset of the overlay.



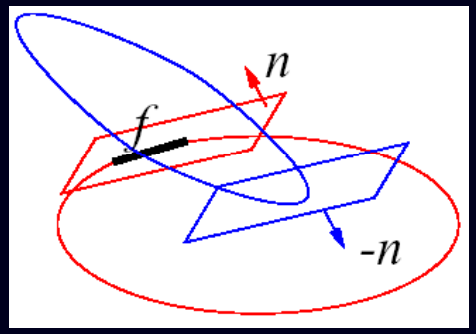


Initialization

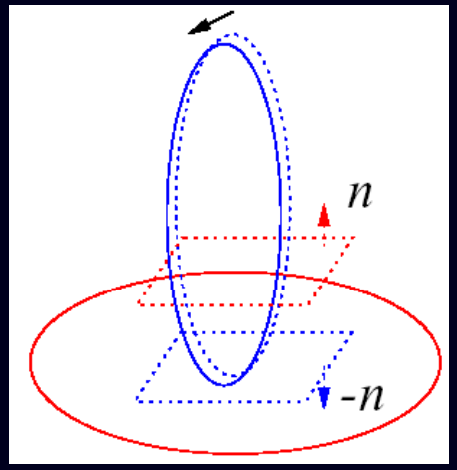
- Guessing an optimal PD direction



Centroid Difference

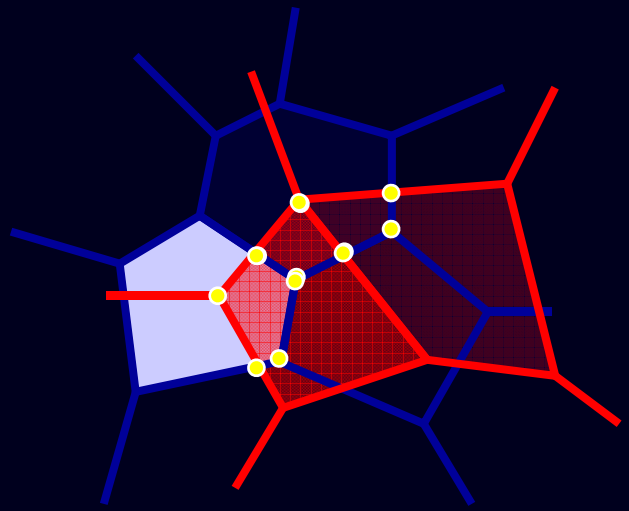


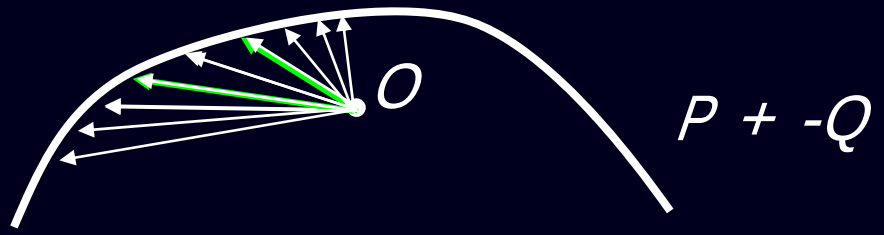
Penetration Witness Feature



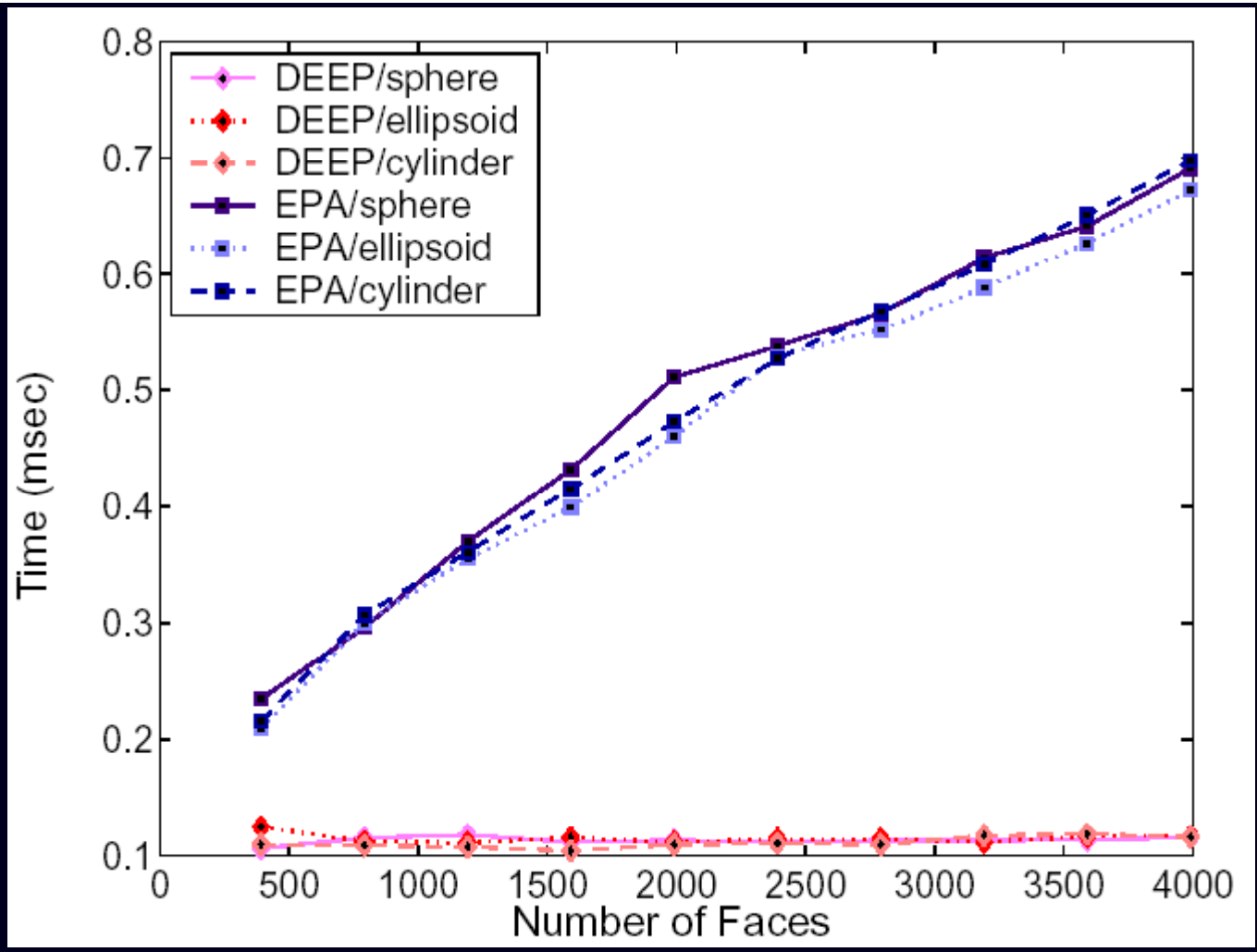
Motion Coherence

Iteration

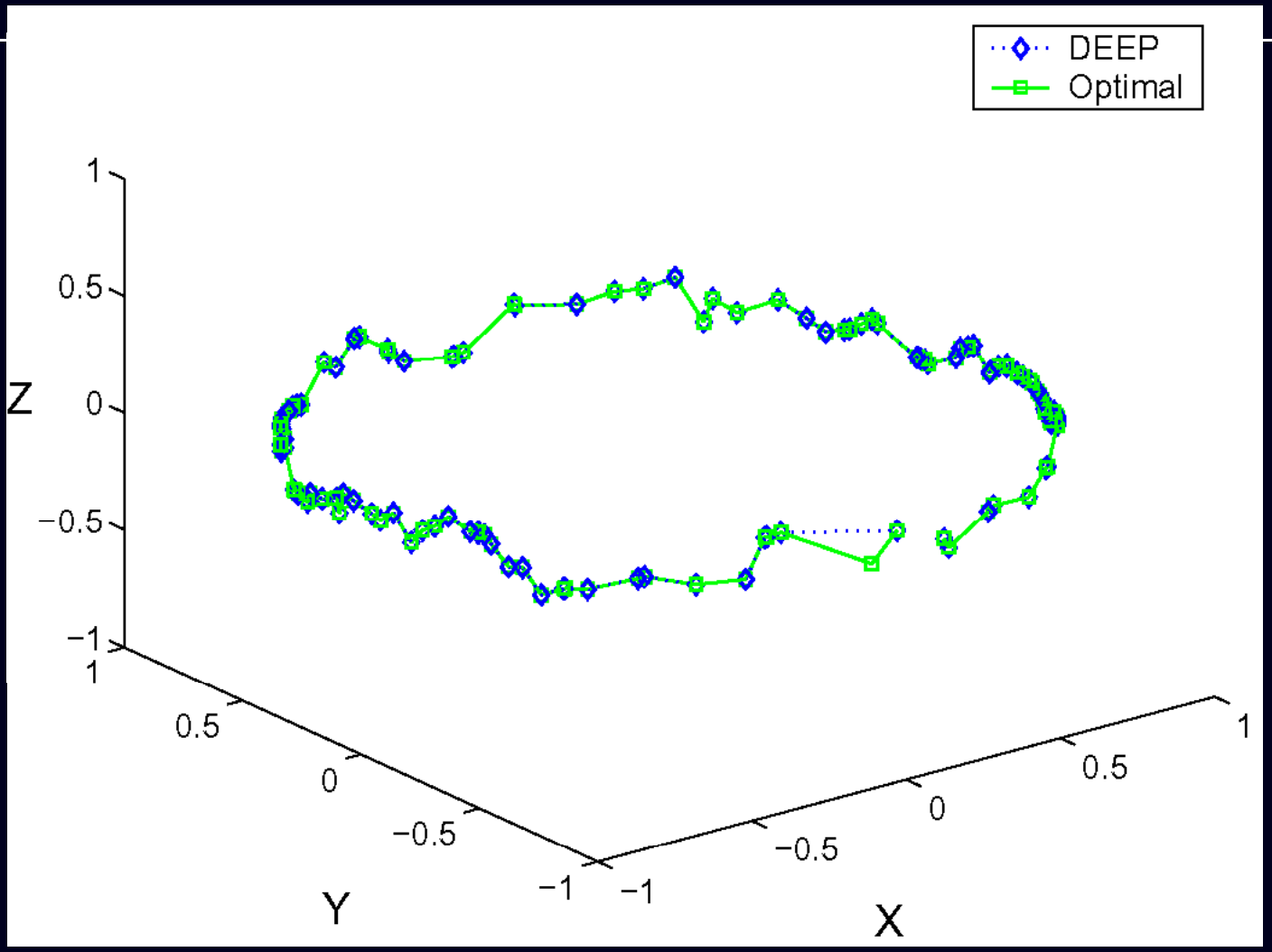




Timings (Fixed PD)



PD Direction Tracking (DEEP)



Softwares

- SWIFT, SWIFT++, RAPID, PIVOT, PQP, DEEP (distance, collision, penetration depth)
  - http://gamma.cs.unc.edu/collide/packages.shtml
- FAST, CATCH (continuous collision detection)
  - http://graphics.ewha.ac.kr

References

[EL00] S. A. Ehmann and M. C. Lin, Accelerated Proximity Queries Between Convex Polyhedra By Multi-Level Voronoi Marching. In *Proc. ICRA*, 2000

[EL01] Stephen A. Ehmann and Ming C. Lin, Accurate and Fast Proximity Queries between Polyhedra Using Surface Decomposition, in *Proc. of Eurographics*, 2001

[GLM96] S. Gottschalk, M. C. Lin and D. Manocha, *OBB-Tree: A Hierarchical Structure for Rapid Interference Detection*, Proc. of ACM SIGGRAPH 1996

[ZLK06] X. Zhang, M. Lee and Y. J. Kim, Interactive Continuous Collision Detection for Non-Convex Polyhedra, Proc. of Pacific Graphics 2006

[KLM04] Y. J. Kim, M. C. Lin and D. Manocha, Incremental Penetration Depth Estimation between Convex Polytopes using Dual-space Expansion, IEEE TVCG Mar. 2004

Thank You!

Visit http://graphics.ewha.ac.kr for more info

Next stop: contact dynamics

Ordinary Differential Equations (ODE)

Young J. Kim

http://graphics.ewha.ac.kr

Dept of CSE

Ewha Womans University

Differential Equations

- Composed of functions and their derivatives

$$e.g. \quad \frac{d^2x}{dt^2} = \frac{f(x)}{m}$$ Newton's second law

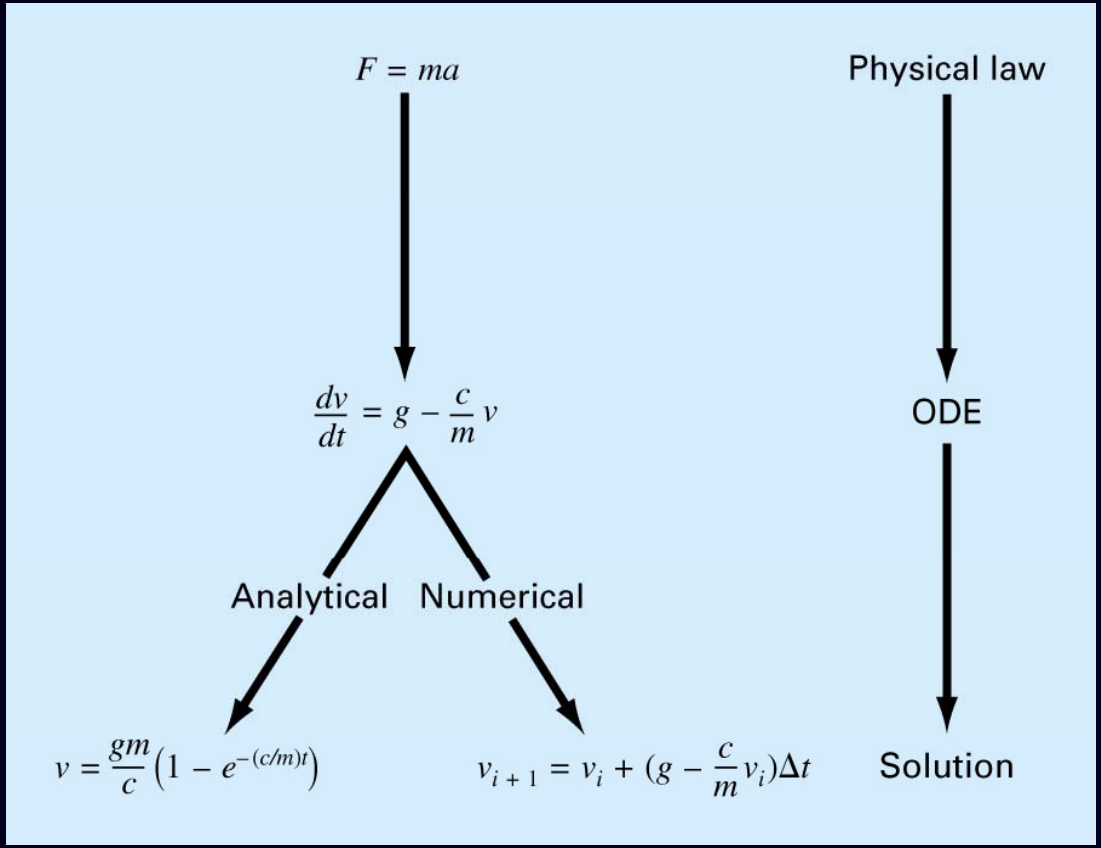
  - Ordinary differential equation (ODE)
    - One independent variable

  - Partial differential equation (PDE)
    - More than one independent variable

$$e.g. \quad \frac{\partial u(t,x)}{\partial t} = \alpha \frac{\partial^2 u(t,x)}{\partial x^2}$$ Heat equation

Differential Equations

- Fundamental in engineering



Runge-Kutta Methods

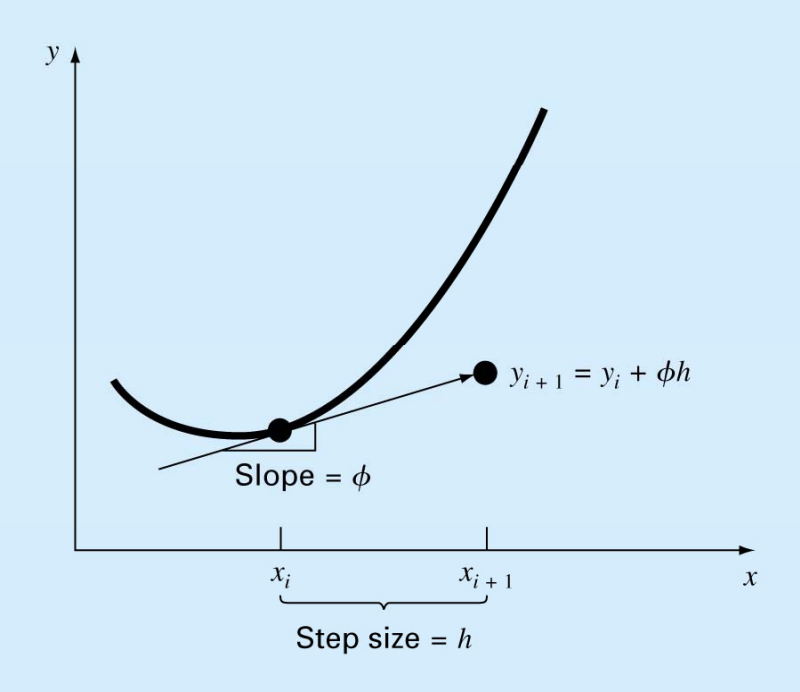
Carl Runge

(1856-1929, Germany)

Numerical Methods for ODE

- Solve $\frac{dy}{dx} = f(x, y)$

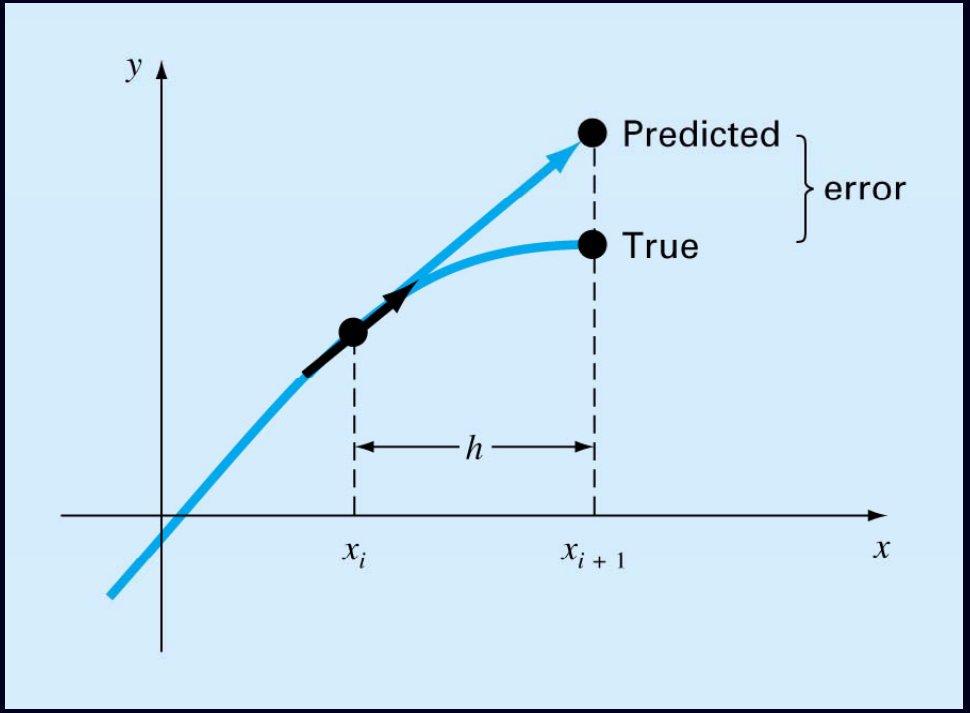
1. Euler
2. Midpoint
3. Runge-Kutta (RK2, RK3, RK4)



Euler's Method

$$\frac{dy}{dx} = f(x, y) = \varphi$$

$$y_{i+1} = y_i + f(x_i, y_i)h$$



Example

- Find y from x=0 to 1 with h=0.5

$$\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5, \quad (x, y) = (0,1)$$

$$y = -0.5x^4 + 4x^3 - 10x^2 + 8.5x + 1$$

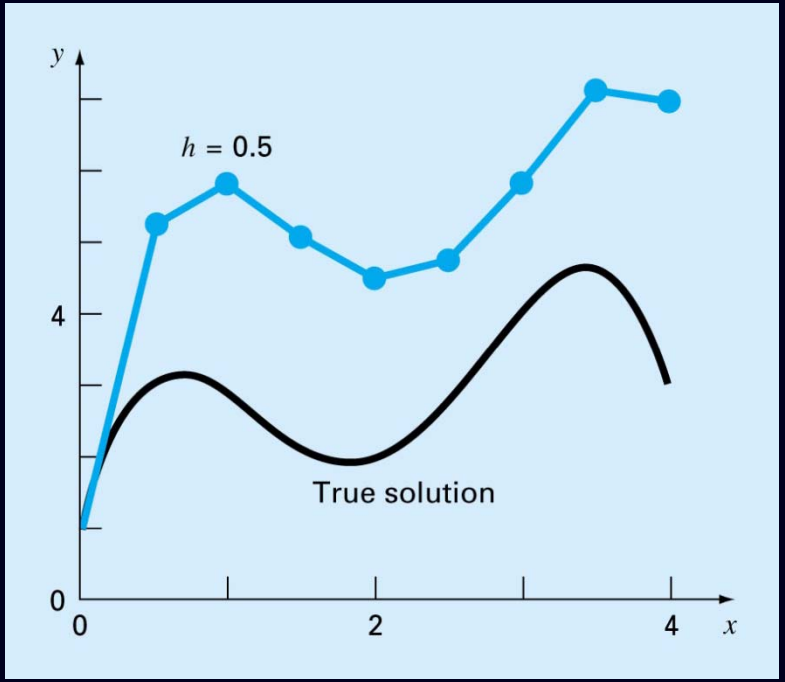
$$y(0.5) = y(0) + f(0,1)0.5 = 1 + 8.5(0.5) = 5.25$$

$$f(0,1) = -2(0)^3 + 12(0)^2 - 20(0) + 8.5 = 8.5$$

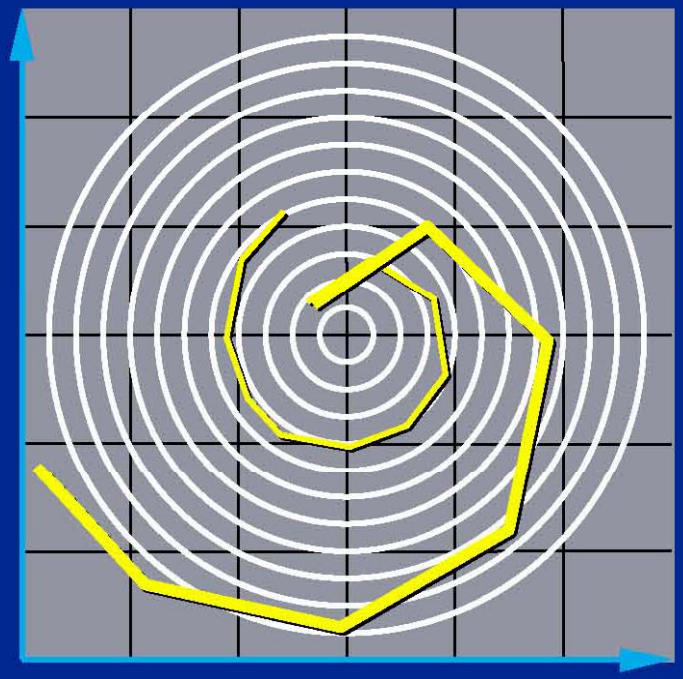
$$y(1) = y(0.5) + f(0.5, 5.25)0.5$$

$$= 5.25 + [-2(0.5)^3 + 12(0.5)^2 - 20(0.5) + 8.5]0.5$$

$$= 5.875$$

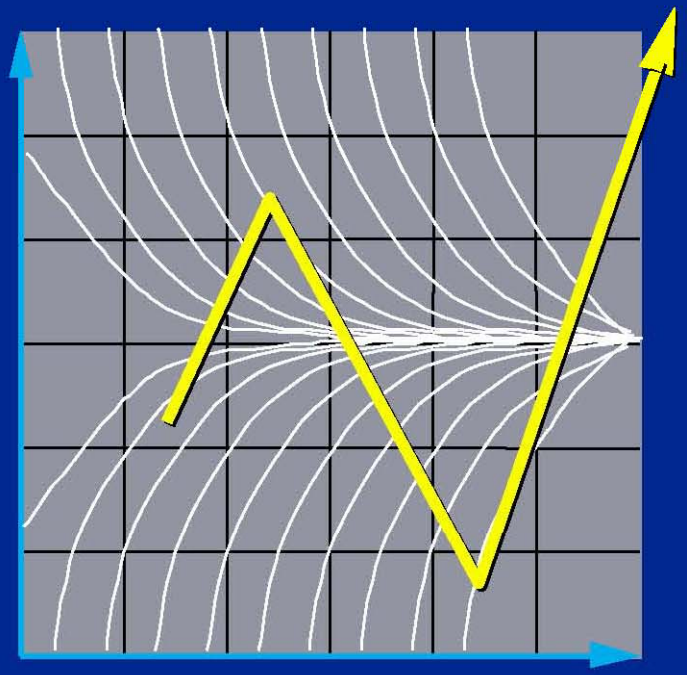


Problem I: Inaccuracy



Error turns x(t) from a circle into the spiral of your choice.

Problem II: Instability

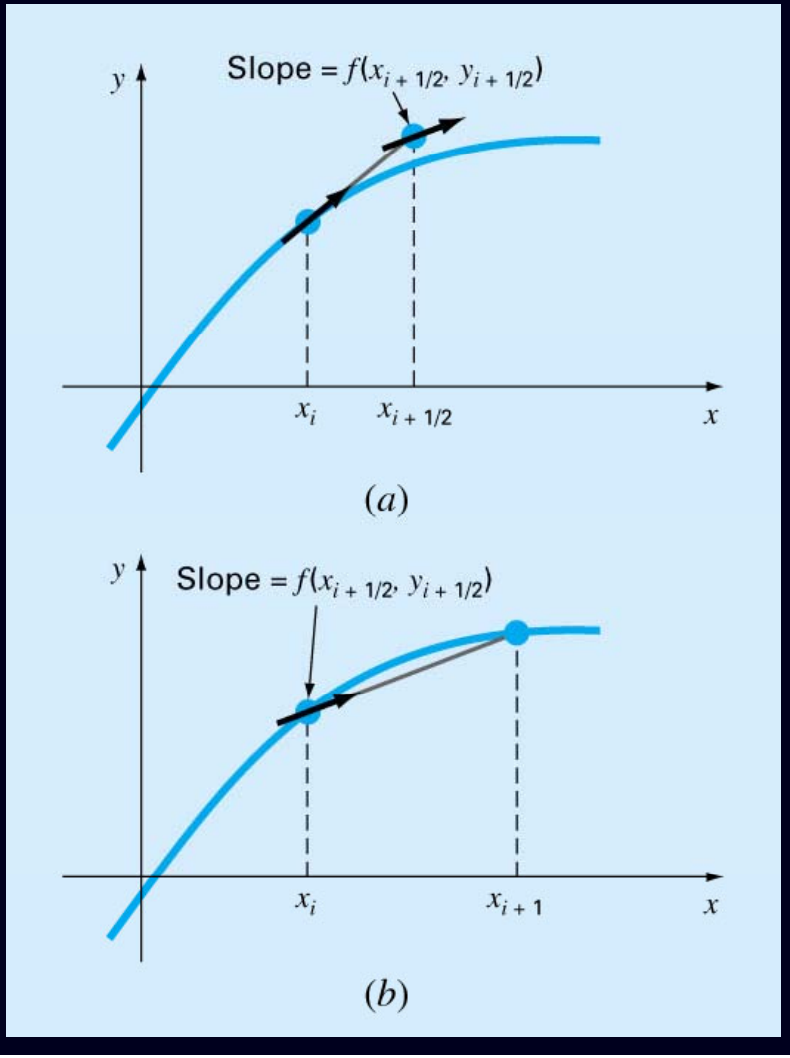


to Neptune!

The Mid-point Method

$$y_{i+\frac{1}{2}} = y_i + f(x_i, y_i)\frac{h}{2}$$

$$y_{i+1} = y_i + f(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}})h$$



Example

- Find y from x=0 to 1 with h=0.5 using mid-point

$$\frac{dy}{dx} = f(x,y) = -2x^3 + 12x^2 - 20x + 8.5, \quad (x,y) = (0,1)$$

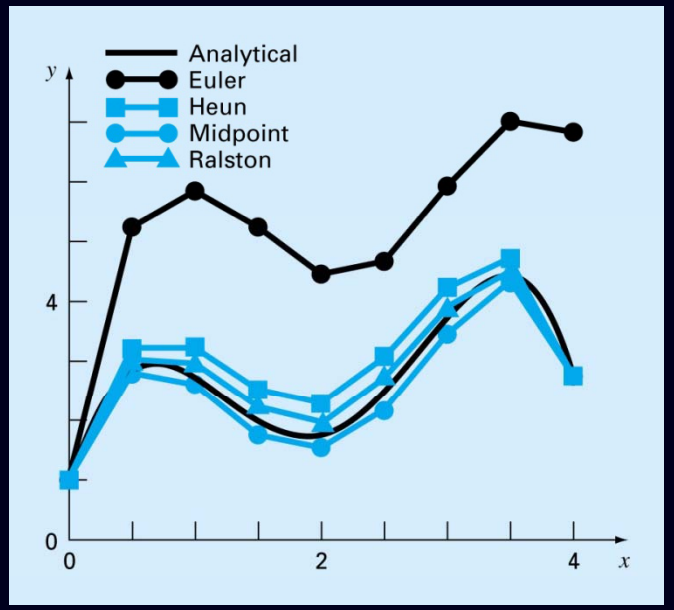
$$y_{\frac{1}{2}} = y_0 + f(x_0, y_0)\frac{0.5}{2} = 1 + 8.5\frac{0.5}{2} = 3.125$$

$$y_1 = 1 + f(x_{\frac{1}{2}}, y_{\frac{1}{2}}) \times 0.5 = 1 + f(0 + \frac{1}{2} \times 0.5, 3.125) \times 0.5$$

$$= 1 + \{-2(0.25)3 + 12(0.25)2 - 20(0.25) + 8.5\} \times 0.5$$

$$= 1 + 4.21875 \times 0.5$$

$$= 3.109375$$



Runge-Kutta Methods

- Order-n RK

$$y_{i+1} = y_i + \phi(x_i, y_i, h)h$$

$$\phi = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n \qquad e.g.\ \text{Euler(RK1)}: \phi = f(x_i, y_i)$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h)$$

$$k_3 = f(x_i + p_2 h, y_i + q_{21} k_1 h + q_{22} k_2 h)$$

$$\vdots$$

$$k_n = f(x_i + p_{n-1} h, y_i + q_{n-1,1} k_1 h + q_{n-1,2} k_2 h + \cdots + q_{n-1,n-1} k_{n-1} h)$$

Second-order Runge Kutta

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2) h$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h)$$

$$a_1 + a_2 = 1$$

$$a_2 p_1 = \frac{1}{2}$$

$$a_2 q_{11} = \frac{1}{2}$$

RK2 Proof

$$y_{i+1} = y_i + (a_1 f(x_i, y_i) + a_2 f(x_i + p_1 h, y_i + q_{11} k_1 h))h$$

$$y_{i+1} = y_i + f(x_i, y_i)h + \frac{f'(x_i, y_i)}{2!}h^2$$

$$f'(x_i, y_i) = \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y}\frac{dy}{dx}$$

Chain rule

$$y_{i+1} = y_i + f(x_i, y_i)h + \left(\frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y}\frac{dy}{dx}\right)\frac{h^2}{2!}$$

$$g(x+r, y+s) = g(x,y) + r\frac{\partial g}{\partial x} + s\frac{\partial g}{\partial y} + \frac{1}{2!}\left[r^2\frac{\partial^2 f}{\partial u^2} + 2\frac{\partial f}{\partial u}\frac{\partial f}{\partial v}rs + s^2\frac{\partial^2 f}{\partial v^2}\right] + \cdots$$

Taylor expansion for two variables

$$f(x_i + p_1 h, y_i + q_{11} k_1 h) = f(x_i, y_i) + p_1 h\frac{\partial f}{\partial x} + q_{11} k_1 h\frac{\partial f}{\partial y} + O(h^2)$$

$$y_{i+1} = y_i + a_1 h f(x_i, y_i) + a_2 h f(x_i, y_i) + a_2 p_1 h^2\frac{\partial f}{\partial x} + a_2 q_{11} k_1 h^2\frac{\partial f}{\partial y} + O(h^3)$$

$$= y_i + [a_1 f(x_i, y_i) + a_2 f(x_i, y_i)]h + \left[a_2 p_1\frac{\partial f}{\partial x} + a_2 q_{11} k_1\frac{\partial f}{\partial y}\right]h^2 + O(h^3)$$

$$\therefore \begin{cases} a_1 + a_2 = 1 \\ a_2 p_1 = \frac{1}{2} \\ a_2 q_{11} = \frac{1}{2} \end{cases}$$

Midpoint Method (RK2)

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h$$

$$a_1 + a_2 = 1, a_2 p_1 = \frac{1}{2}, a_2 q_{11} = \frac{1}{2}$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + p_1 h, y_i + q_{11} k_1 h)$$

$$a_2 = 1, a_1 = 0, p_1 = \frac{1}{2}, q_{11} = \frac{1}{2}$$

$$y_{i+1} = y_i + f(x_i + \frac{1}{2}h, y_i + \frac{1}{2} f(x_i, y_i)h)h$$

Fourth-order Runge Kutta

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)h$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h)$$

$$k_3 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2 h)$$

$$k_4 = f(x_i + h, y_i + k_3 h)$$

Example

- Perform one step of RK4 with h=0.5

$$\frac{dy}{dx} = f(x,y) = 4e^{0.8x} - 0.5y, \quad (x,y) = (0,2)$$

$$k_1 = f(0,2) = 4e^{0.8(0)} - 0.5(2) = 3$$

$$k_2 = f(0+\frac{1}{2}(0.5), 2+\frac{1}{2}3(0.5)) = f(0.25, 2.75) = 4e^{0.8(0.25)} - 0.5(2.75) = 3.510611$$

$$k_3 = f(0+\frac{1}{2}(0.5), 2+\frac{1}{2}3.510611(0.5)) = f(0.25, 2.877653) = 3.446785$$

$$k_4 = f(0.5, 2+3.446785(0.5)) = 4.105603$$

$$\phi = \frac{1}{6}[3+2(3.510611)+2(3.446785)+4.105603] = 3.503399$$

$$\therefore y_1 = y(0.5) = 2+3.503399(0.5) = 3.751699$$

Example

- Perform one step of RK4 with h=0.5

$$\frac{dy}{dx} = f(x,y) = -2x^3 + 12x^2 - 20x + 8.5, \quad (x,y) = (0,1)$$

$$\frac{dy}{dx} = -2x^3 + 12x^2 - 20x + 8.5, \quad (x,y) = (0,1)$$

$$k_1 = 8.5, k_2 = 4.21875, k_3 = 4.21875, k_4 = 1.25$$

$$y_1 = 1 + \left\{\frac{1}{6}[8.5 + 2(4.21875) + 2(4.21875) + 1.25]\right\}0.5$$

$$= 3.21875$$

$$= \text{exact solution}$$

- RKn is exact upto polynomials of degree n

Error Analysis of RK

- Order-n RK: $O(h^n)$
  - Euler: $O(h)$
  - Mid-point (RK2): $O(h^2)$
  - RK4: $O(h^4)$

NEXT STOP: 5 MIN BREAK