DIRECT SOLVER

STIFFNESS

IMPLICIT SOLVER

HYBRID

CONTINUMM MECHANCIS

OPTIMIZATIONS

APIC

PBD

NEWTON-RAPHSON

GRADIENT

GAUSS-SEIDEL

XPBD

FEM

FLIP

∇f

PDE

ENERGY MINIMIZATIONS

Δf

YOUNGS MODULUS

JACOBI

DISCRETE SHELL

MPM

LAGRANGIAN MODEL

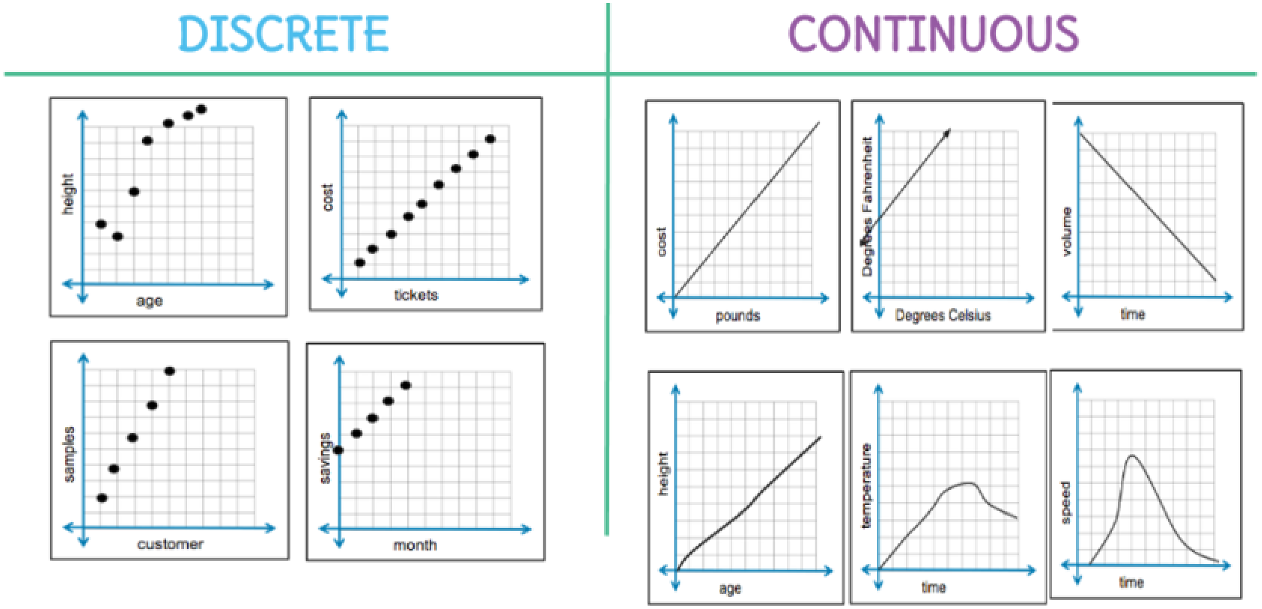
EULERIAN MODEL

SPH

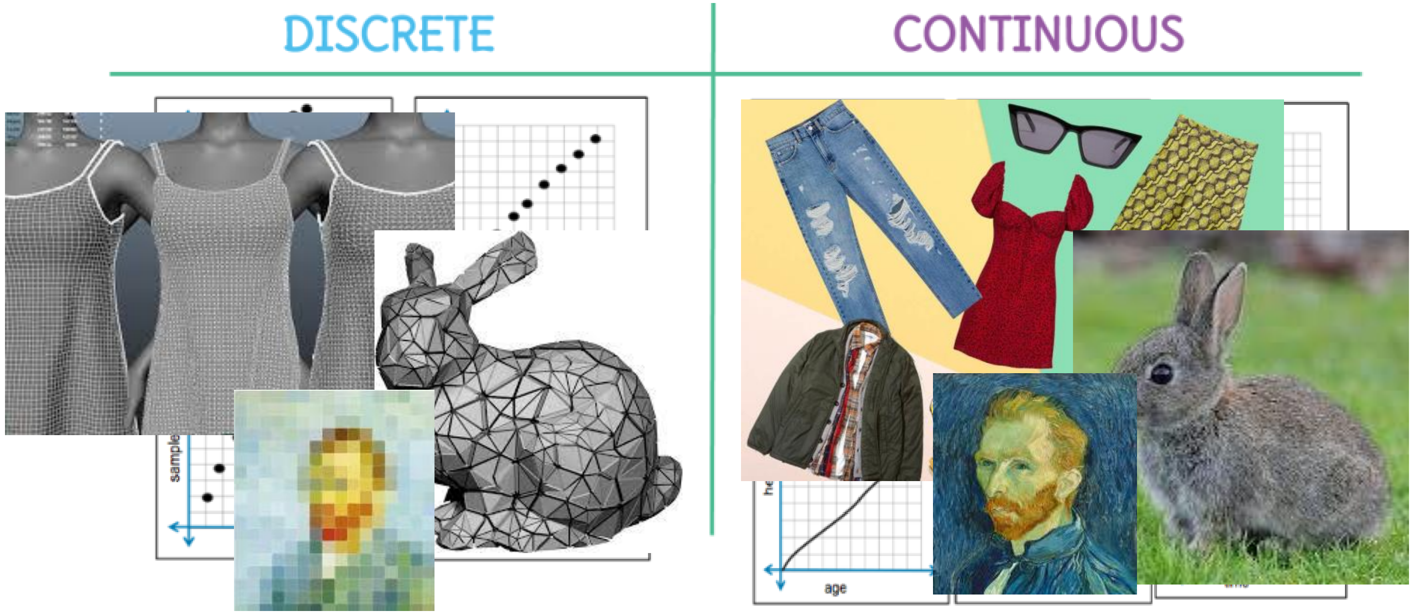
EXPLICIT SOLVER

BEFORE THAT ….

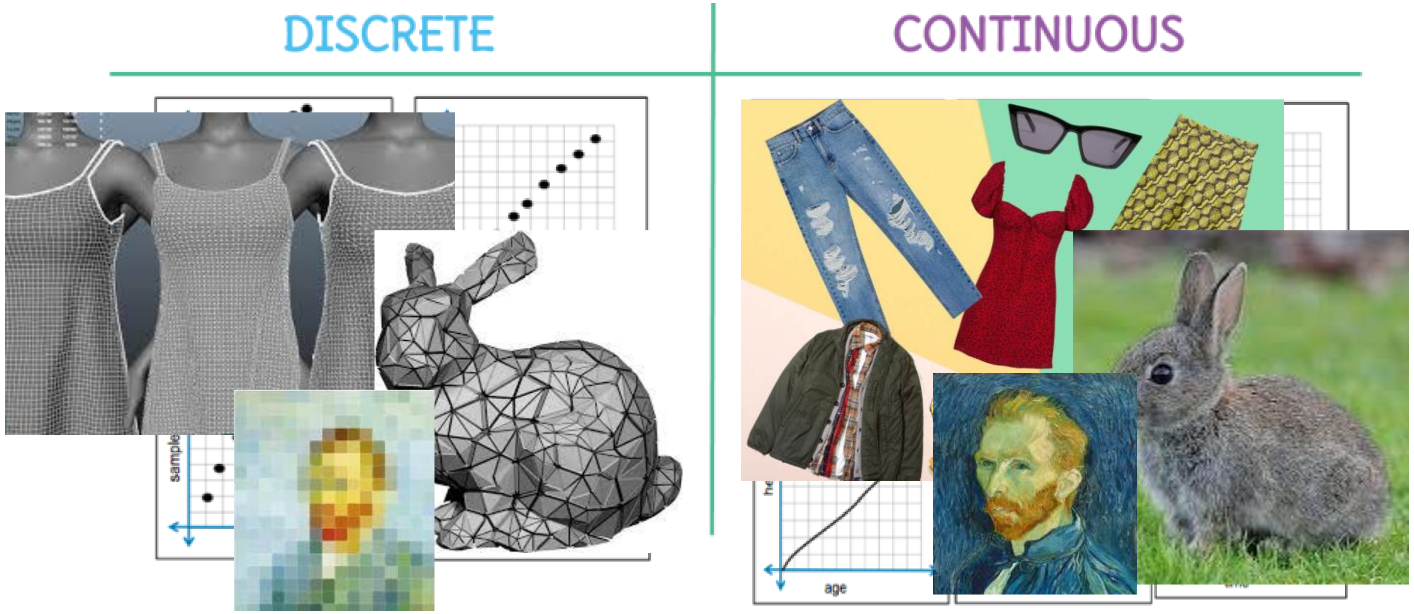
DISCRETE vs CONTINUOUS



DISCRETE vs CONTINUOUS



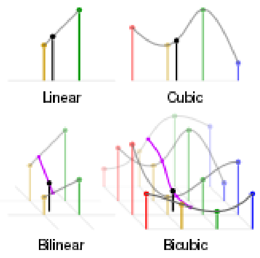
DISCRETE vs CONTINUOUS



DISCRETIZATION

DISCRETE vs CONTINUOUS

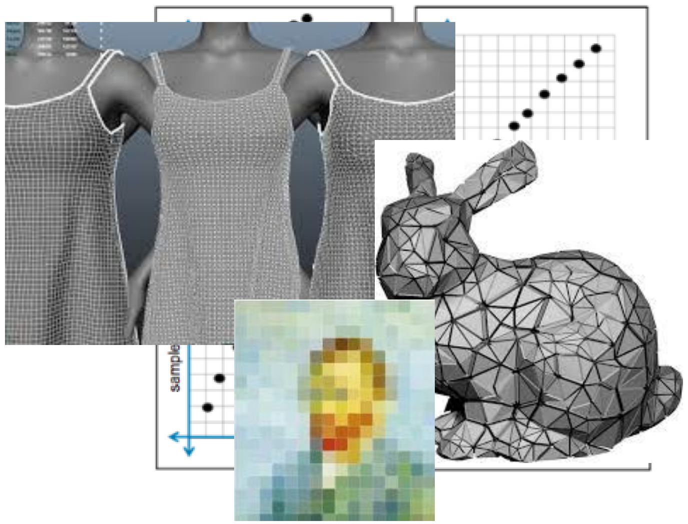
INTERPOLATION

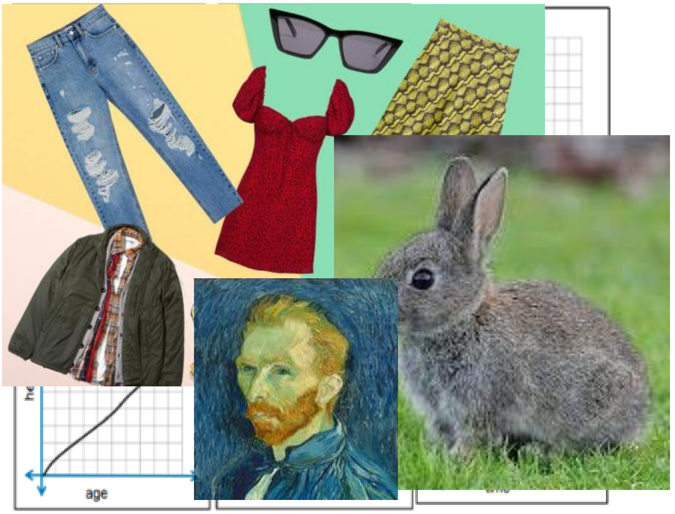


Differentiable : $\nabla f, \Delta f, \frac{\partial u}{\partial x}, ..$

DISCRETE

CONTINUOUS





DISCRETIZATION

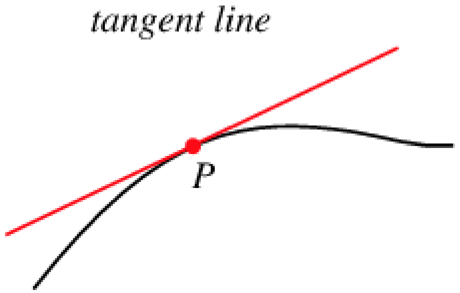
MULTIVARIABLE SCALAR FUNCTION

$$f(x, y, \mathrm{z}, \dots) = \text{a single float}$$

MULTIVARIABLE SCALAR FUNCTION

$$f(x, y, z, \dots) = \text{a single float}$$

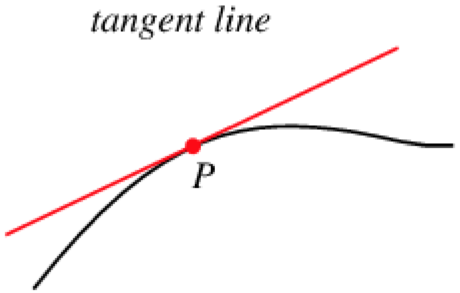
$\dfrac{\partial f}{\partial x}$ => How ***f*** changes when ***x*** changes a tiny bit



MULTIVARIABLE SCALAR FUNCTION

$$f(x, y, \mathrm{z}, \ldots) = \text{a single float}$$

$$\frac{\partial f}{\partial x}$$ => How ***f*** changes when ***x*** changes a tiny bit



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}, \ldots\right]$$ => How ***f*** changes when ***x,y,z,...*** change a tiny bit

MULTIVARIABLE SCALAR FUNCTION

$$f(x, y, \mathrm{z}, \ldots) = \text{a single float}$$

+

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}, \ldots\right]$$

$$f\left(x + \frac{\partial f}{\partial x}, y + \frac{\partial f}{\partial y}, \mathrm{z} + \frac{\partial f}{\partial z}, \ldots\right) =$$

MULTIVARIABLE SCALAR FUNCTION

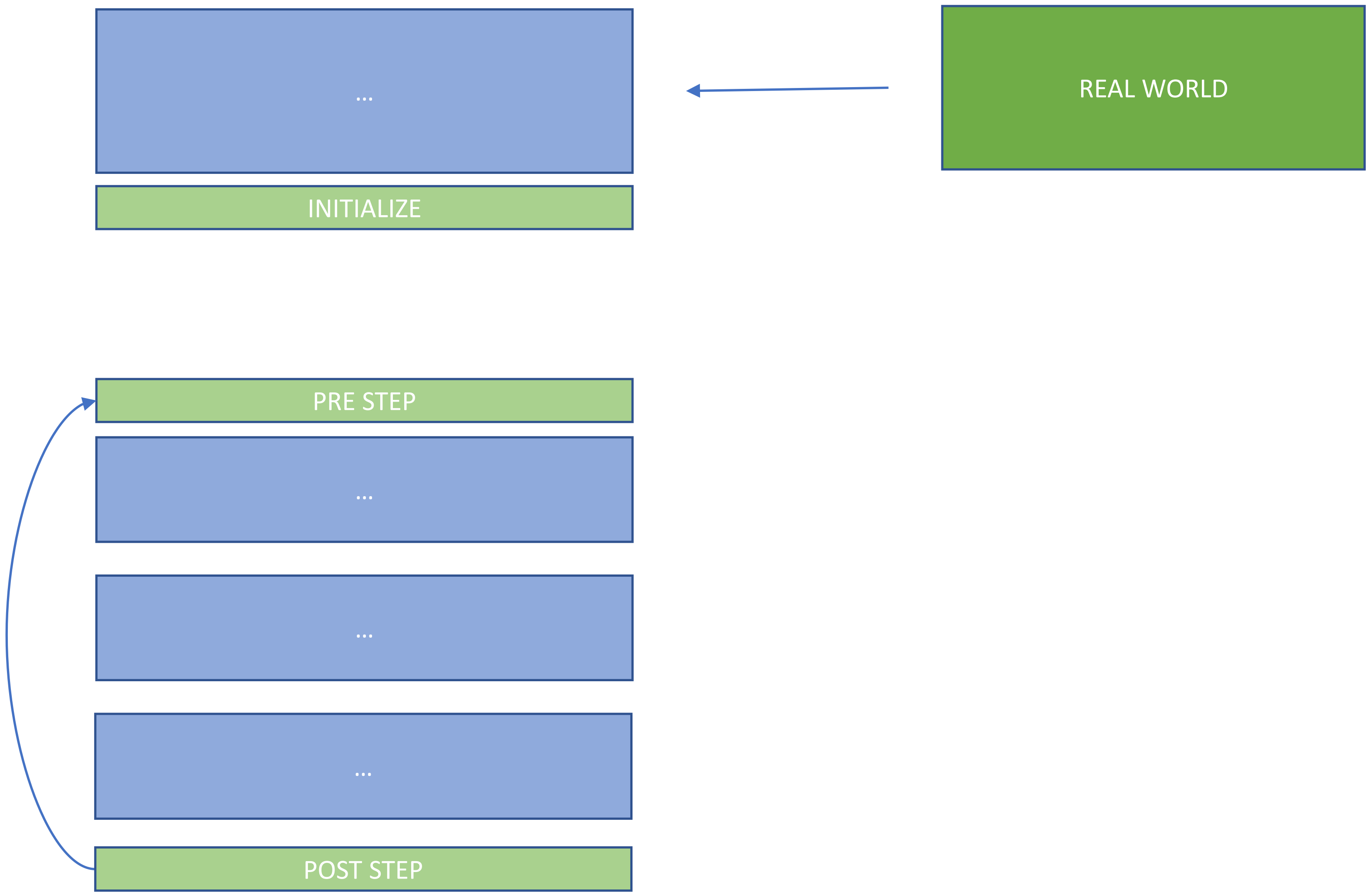
$$f(x, y, \mathrm{z}, \dots) = \text{a single float}$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}, \dots\right]$$

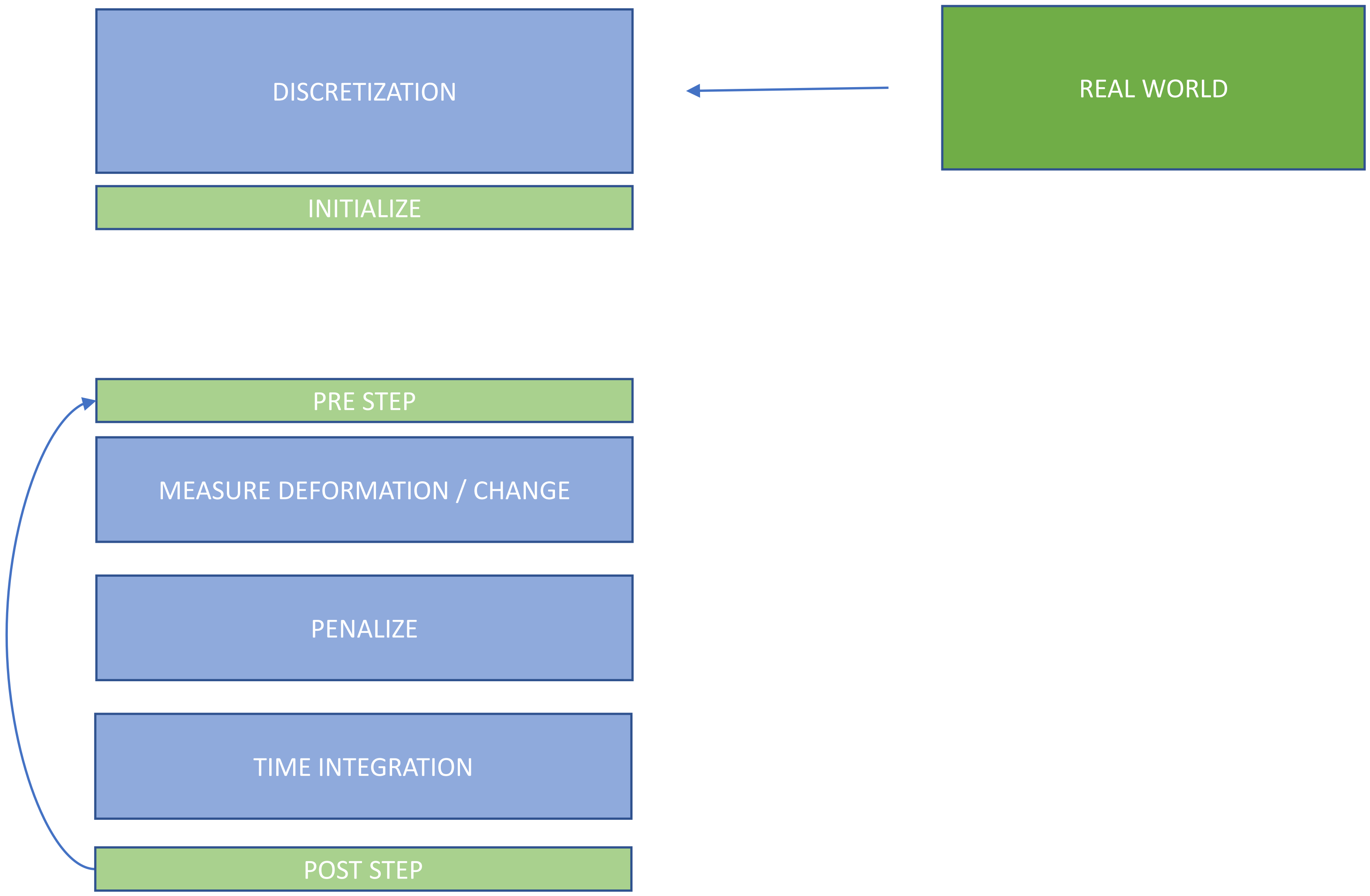
$$f\left(x - \frac{\partial f}{\partial x}, y - \frac{\partial f}{\partial y}, \mathrm{z} - \frac{\partial f}{\partial z}, \dots\right) = \searrow$$

PHYSICS BASED ANIMATION

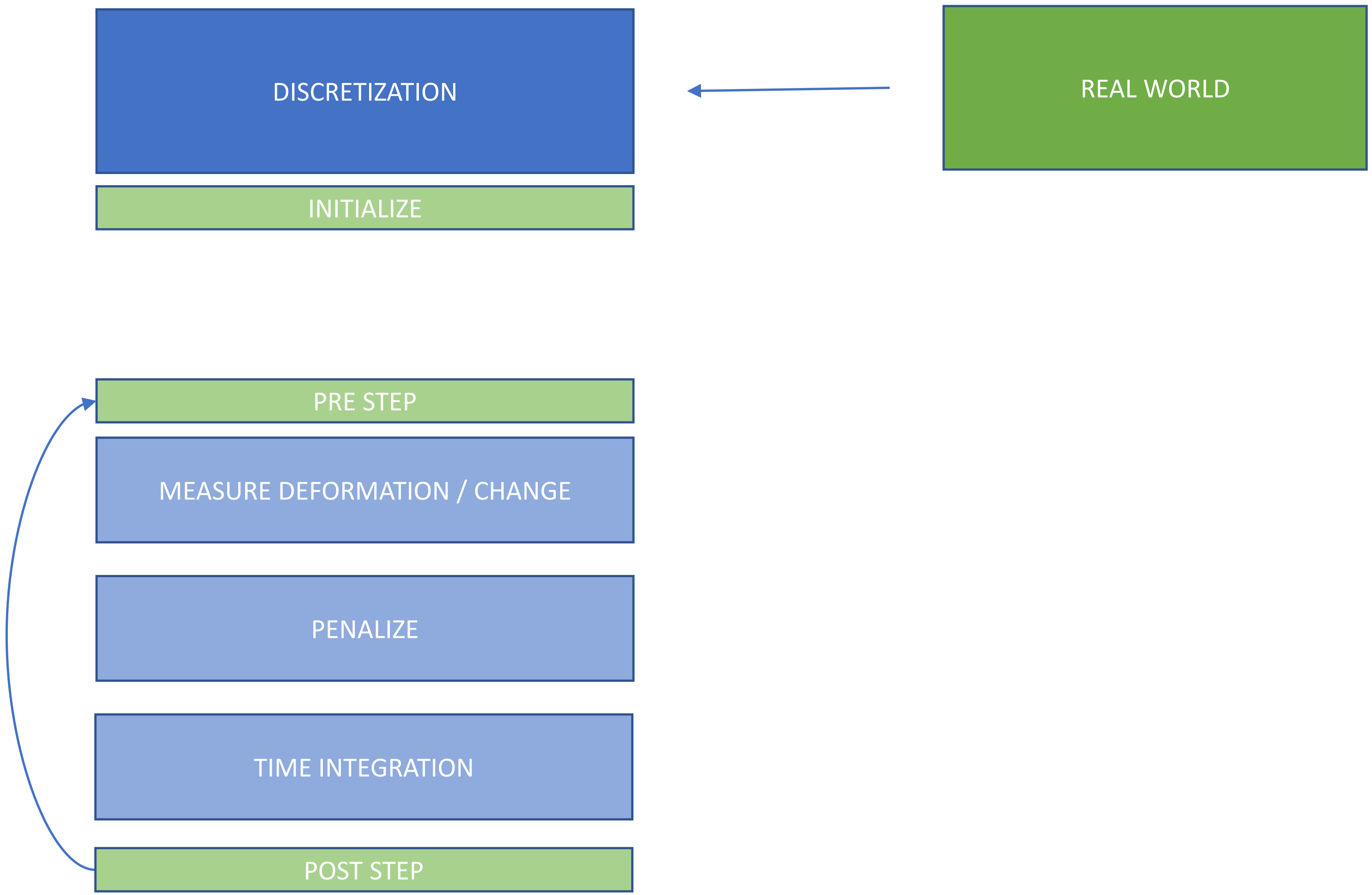
WHAT DO SOLVERS DO



WHAT DO SOLVERS DO



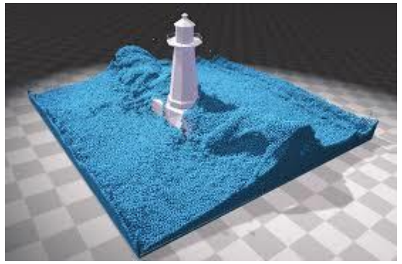
WHAT DO SOLVERS DO



DISCRETIZATION

DISCRETE REPRESENTATION

LAGRANGIAN



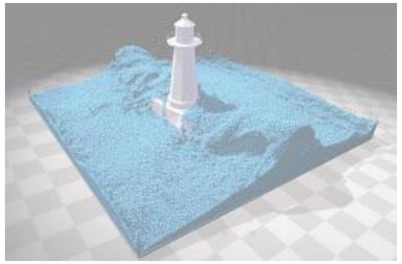
Particle Fluid
Smoothed-particle hydrodynamics (SPH)

Spring Simulation
Position Based Dynamics / Extended PBD

Tetrahedral Simulation
Tetrahedral FEM (Finite Element Method)

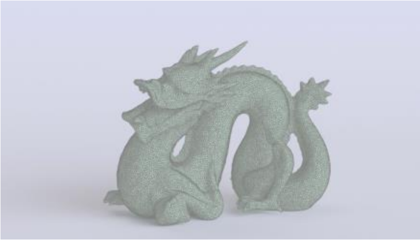
DISCRETE REPRESENTATION

LAGRANGIAN



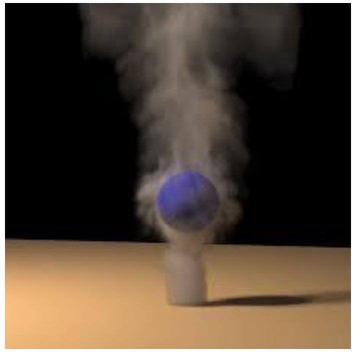
Particle Fluid
Smoothed-particle hydrodynamics (SPH)

Spring Simulation
Position Based Dynamics / Extended PBD



Tetrahedral Simulation
Tetrahedral FEM (Finite Element Method)

EULERIAN

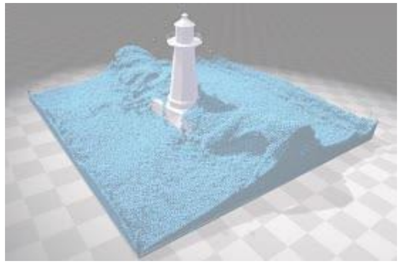


Smoke Simulation
Voxel Based

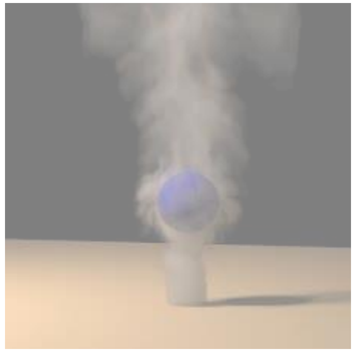
DISCRETE REPRESENTATION

LAGRANGIAN

EULERIAN

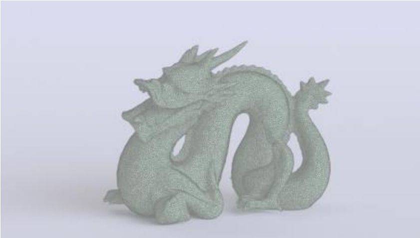


Particle Fluid
Smoothed-particle hydrodynamics (SPH)

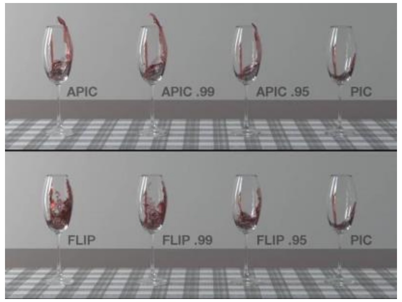


Smoke Simulation
Voxel Based

Spring Simulation
Position Based Dynamics / Extended PBD



Tetrahedral Simulation
Tetrahedral FEM (Finite Element Method)

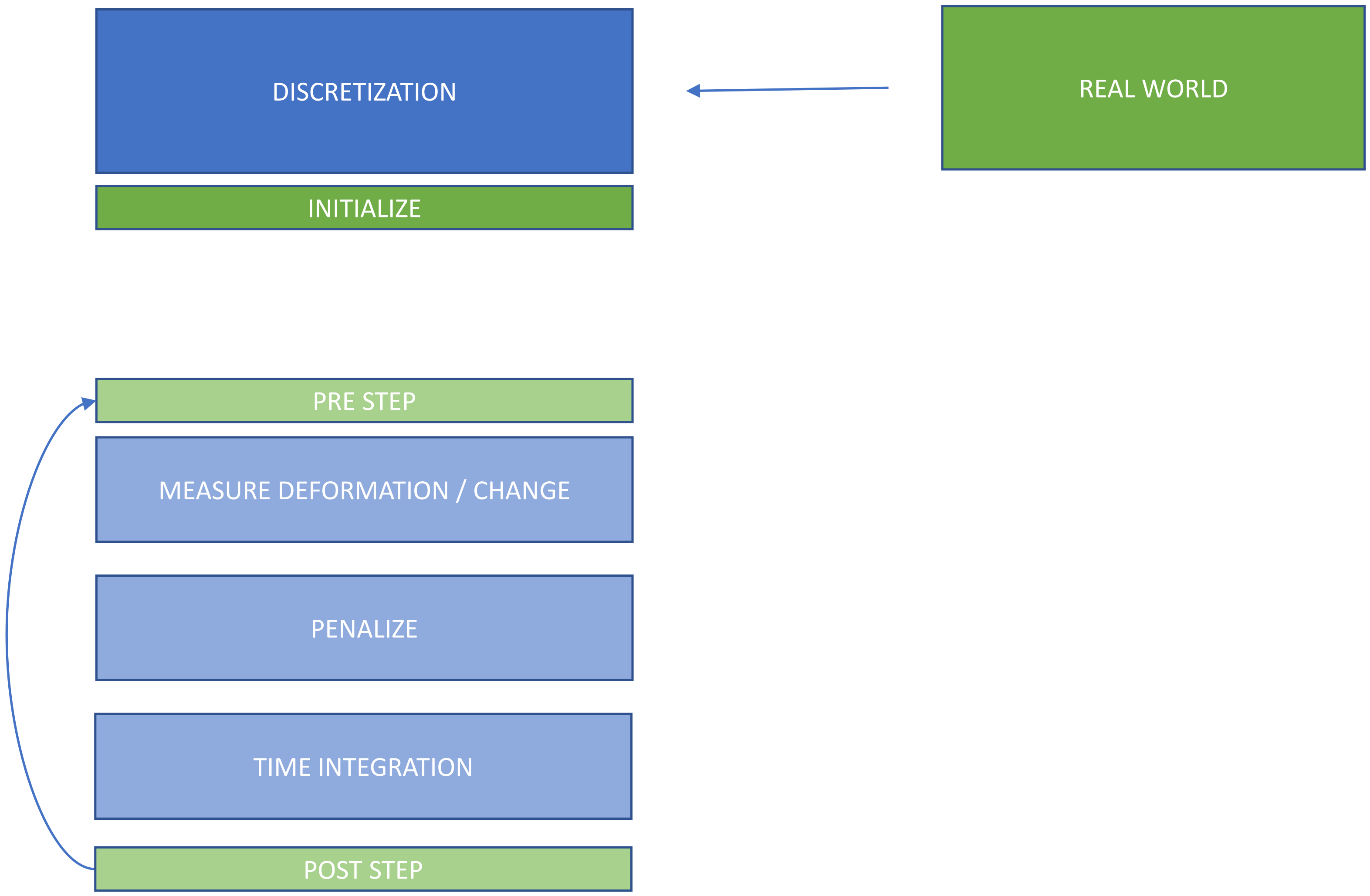


FLIP/PIC/APIC/POLYPIC/…

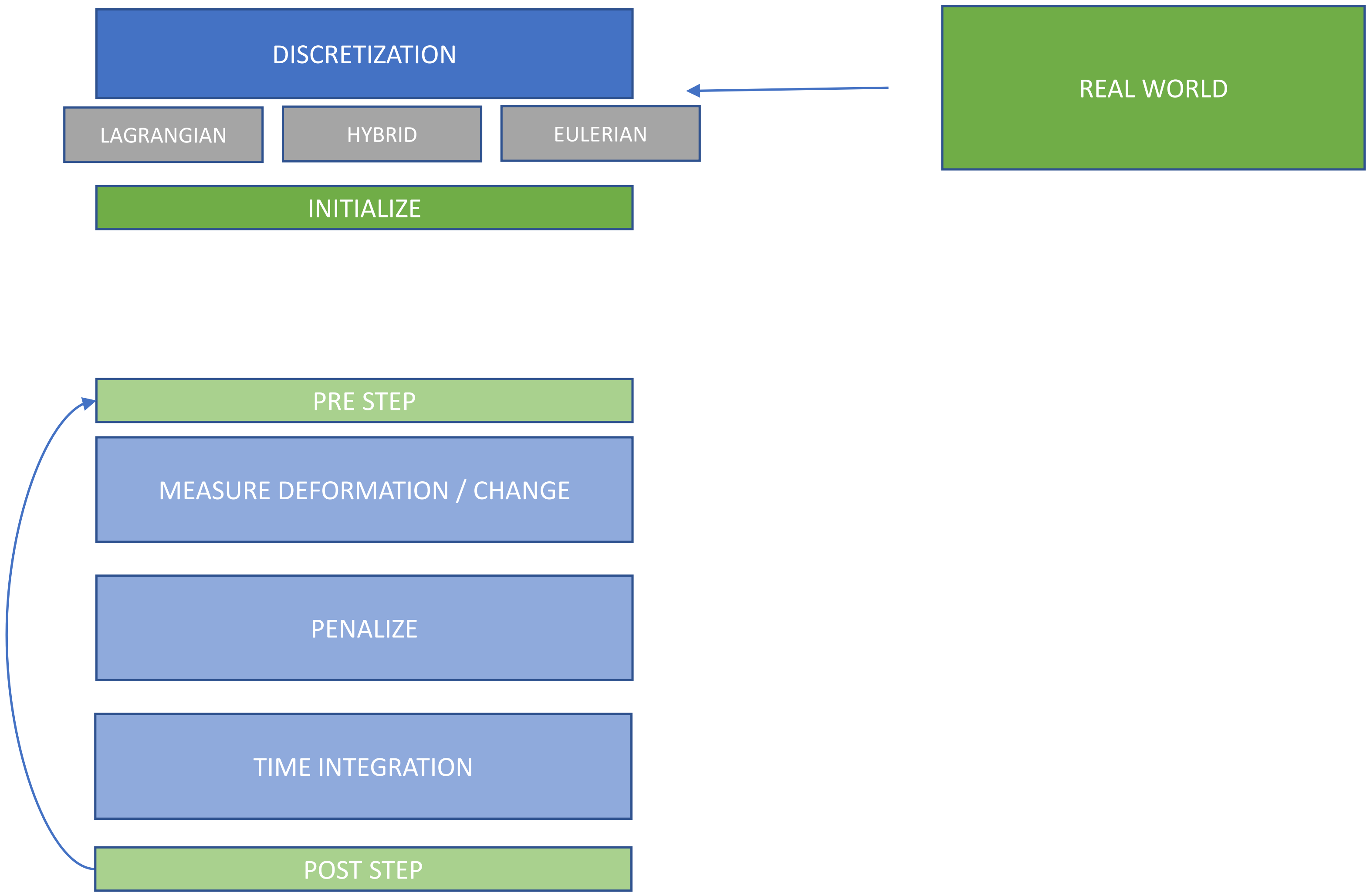
MPM
Material Point Method

HYBRID = LAGRANGIAN + EULERIAN

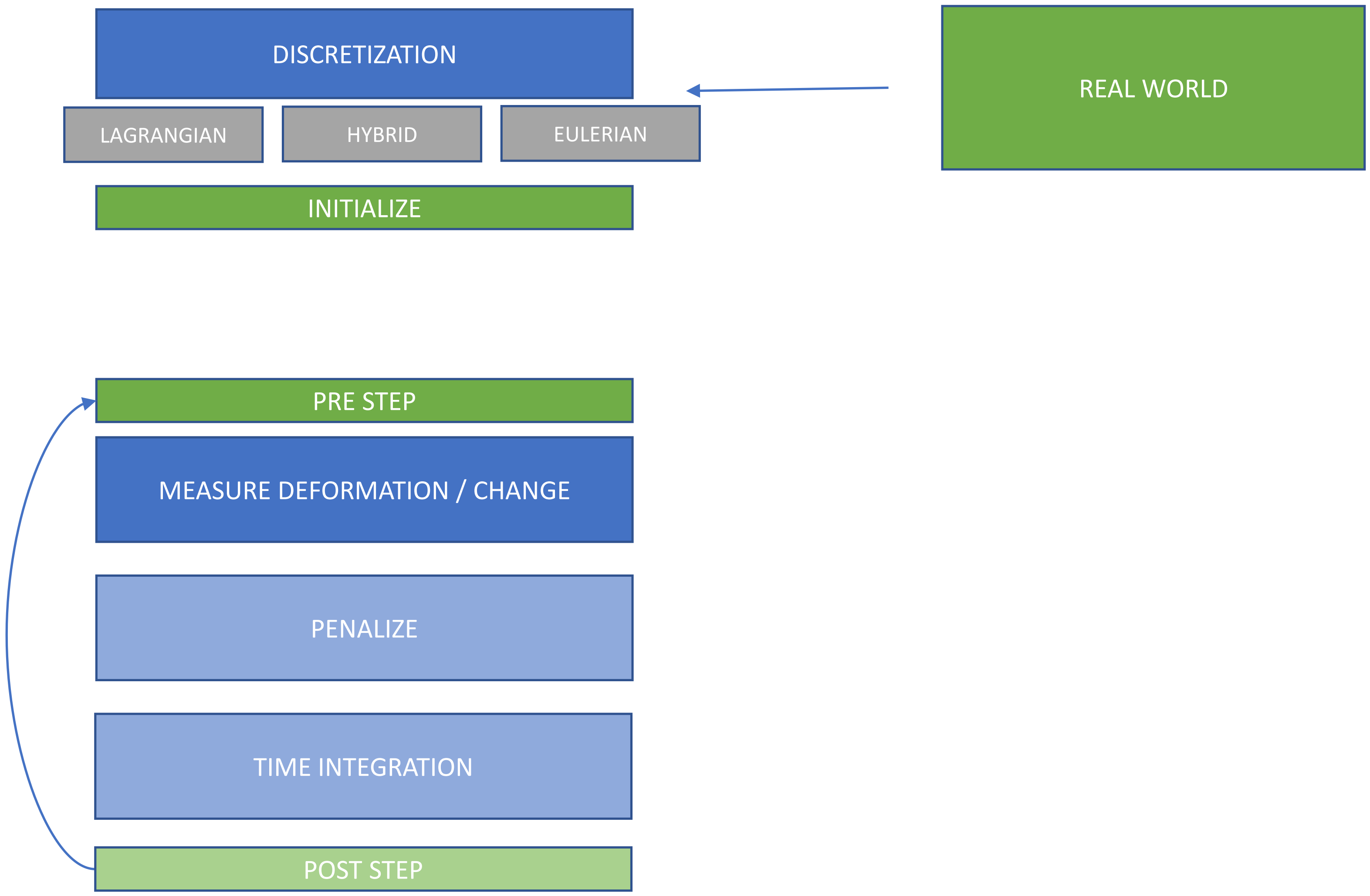
WHAT DO SOLVERS DO



WHAT DO SOLVERS DO



WHAT DO SOLVERS DO



DEFORMATION / CHANGE

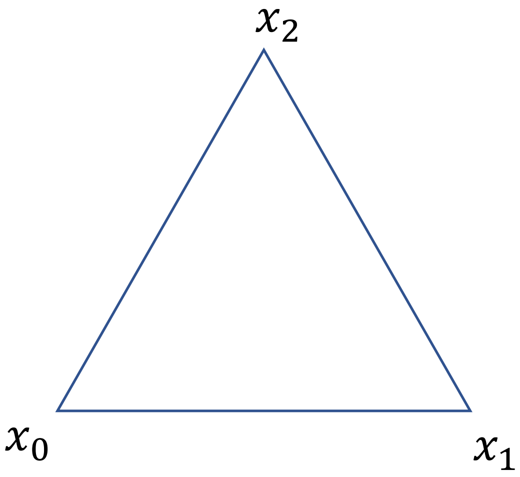
DISCRETE MODEL

GEOMETRIC BASED

Distance

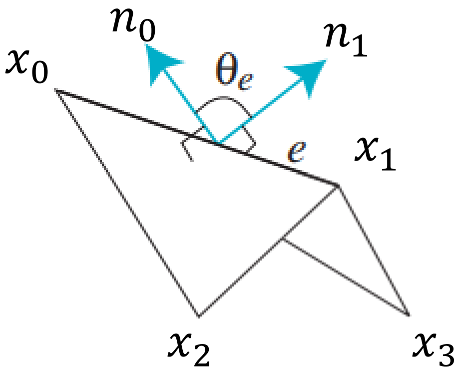
$$e = \|x_1 - x_0\|$$

Area



$$A = \frac{\|(x_1 - x_0) \times (x_2 - x_0)\|}{2}$$

Dihedral Angle

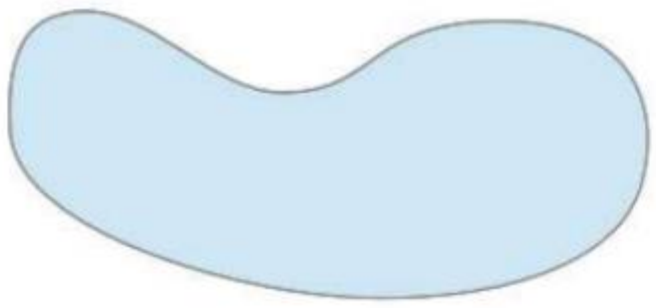


$$\theta = \arctan 2(|n_0 \times n_1|, n_0 \cdot n_1)$$

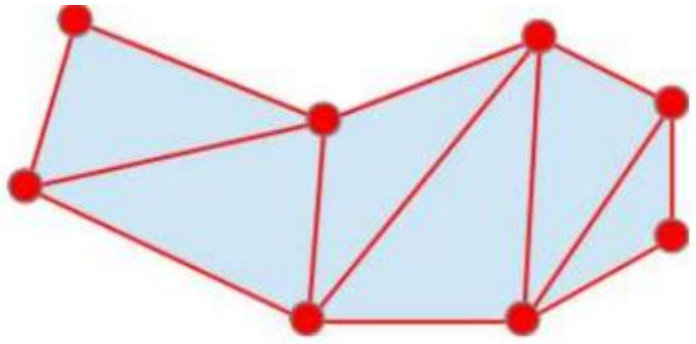
CONTINUOUS MODEL - FEM

CONTINUUM MECHANICS

- GOAL : Use the object as a continuous medium to apply differential calculus and compute an elastic energy



$\simeq$



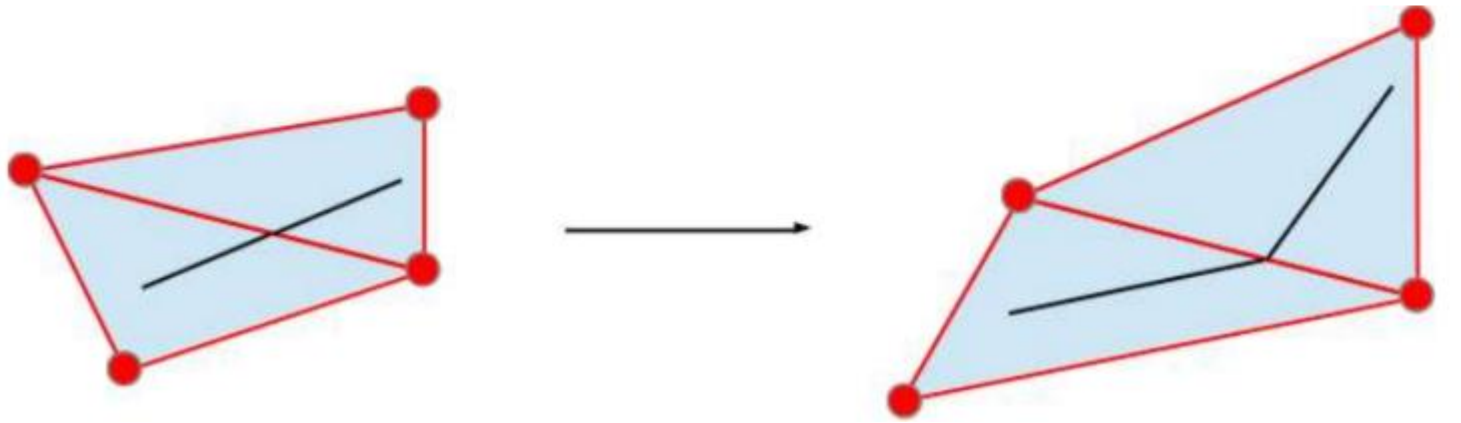
Continuous Object

FEM Object =
Control Nodes + Interpolation Functions (Shape Function)

CONTINUOUS MODEL - FEM

CONTINUUM MECHANICS

- Continuity of deformation depends on the Shape Function (Linear / Non-Linear)
- Build a continuous deformation field : u(x)



Undeformed

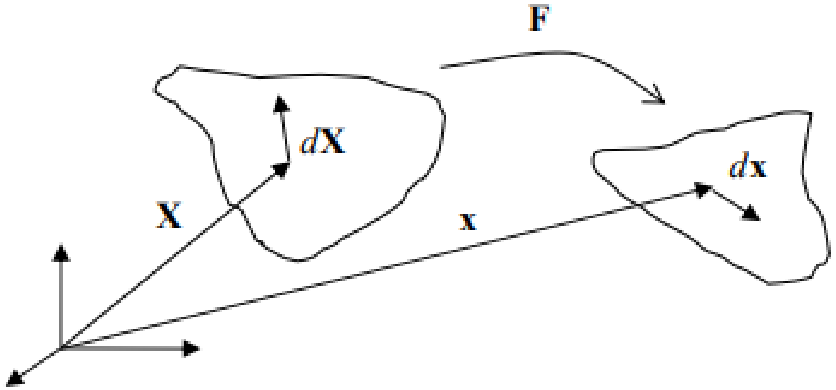
Deformed (Linear FEM)

CONTINUOUS MODEL - DEFORMATION

CONTINUUM MECHANICS – DEFORMATION GRADIENT

Deformation Gradient

$$F = \frac{\partial x_{\mathrm{i}}}{\partial X_{\mathrm{j}}} = \begin{vmatrix} \frac{\partial x}{\mathrm{d}X} & \frac{\partial x}{\partial Y} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} \end{vmatrix}$$



Rest **Deformed**

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

$$u = x - X$$

CONTINUOUS MODEL - DEFORMATION

CONTINUUM MECHANICS – DEFORMATION GRADIENT

Deformation Gradient

$$F = \frac{\partial x_{\mathrm{i}}}{\partial X_{\mathrm{j}}} = \begin{vmatrix} \frac{\partial x}{\mathrm{d}X} & \frac{\partial x}{\partial Y} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} \end{vmatrix}$$

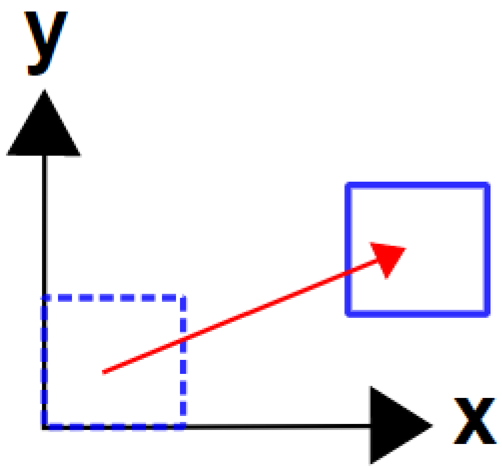
$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

$$u = x - X$$

Rigid Displacement

x=X+5

y=Y+2



$$\boxed{F = \mathrm{I} = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}}$$

CONTINUOUS MODEL - DEFORMATION

CONTINUUM MECHANICS – DEFORMATION GRADIENT

Deformation Gradient

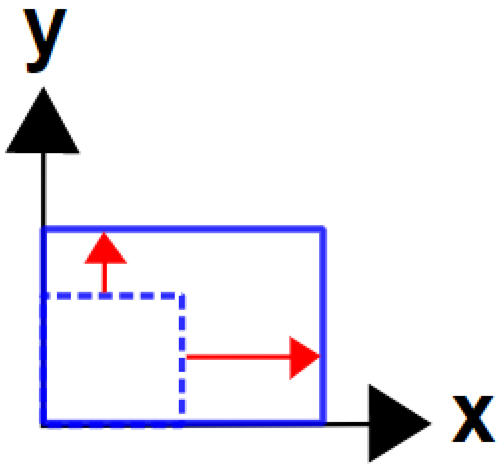
$$F = \frac{\partial x_{\mathrm{i}}}{\partial X_{\mathrm{j}}} = \begin{vmatrix} \frac{\partial x}{\mathrm{d}X} & \frac{\partial x}{\partial Y} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} \end{vmatrix}$$

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

$$u = x - X$$

Stretching

x=2.0X+0Y
y=0X+1.5Y



$$\boxed{F = \begin{vmatrix} 2.0 & 0 \\ 0 & 1.5 \end{vmatrix}}$$

CONTINUOUS MODEL - DEFORMATION

CONTINUUM MECHANICS – DEFORMATION GRADIENT

Deformation Gradient

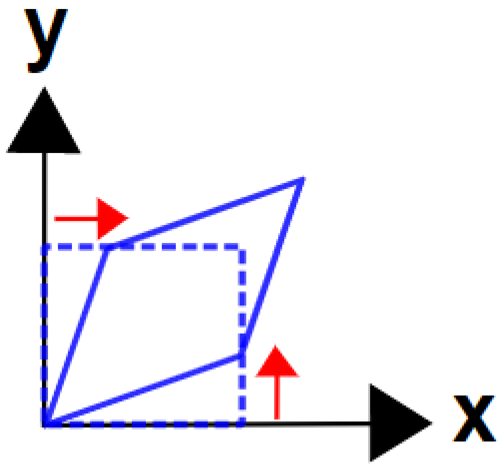
$$F = \frac{\partial x_{\mathrm{i}}}{\partial X_{\mathrm{j}}} = \begin{vmatrix} \frac{\partial x}{\mathrm{d}X} & \frac{\partial x}{\partial Y} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} \end{vmatrix}$$

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

$$u = x - X$$

Shear

x=1.0X+0.5Y
y=0.5X+1.0Y



$$F = \begin{vmatrix} 1.0 & 0.5 \\ 0.5 & 1.0 \end{vmatrix}$$

CONTINUOUS MODEL - DEFORMATION

CONTINUUM MECHANICS – DEFORMATION GRADIENT

Deformation Gradient

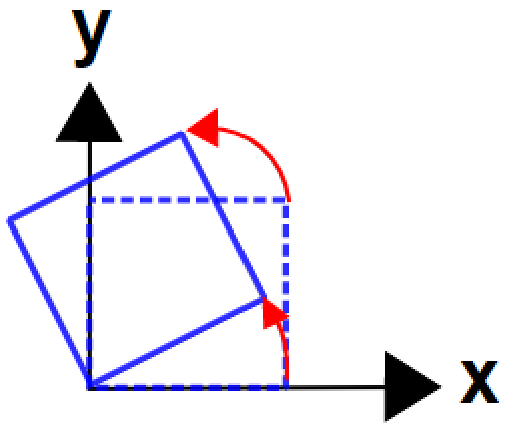
$$F = \frac{\partial x_{\mathrm{i}}}{\partial X_{\mathrm{j}}} = \begin{vmatrix} \frac{\partial x}{\mathrm{d}X} & \frac{\partial x}{\partial Y} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} \end{vmatrix}$$

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

$$u = x - X$$

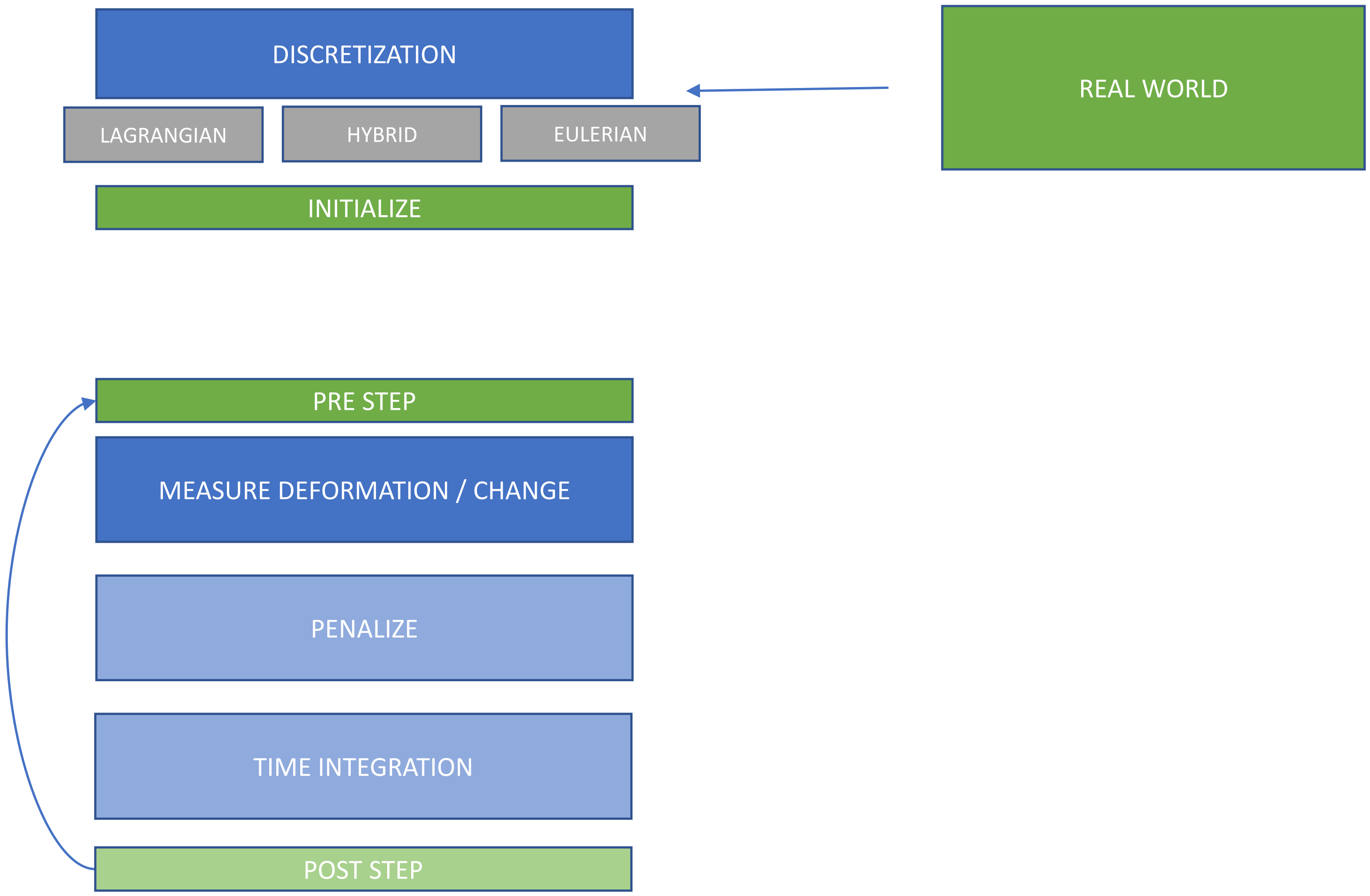
Rigid Rotation

x=X.cos(θ)-Y.sin(θ)
y=X.sin(θ)+Y.cos(θ)

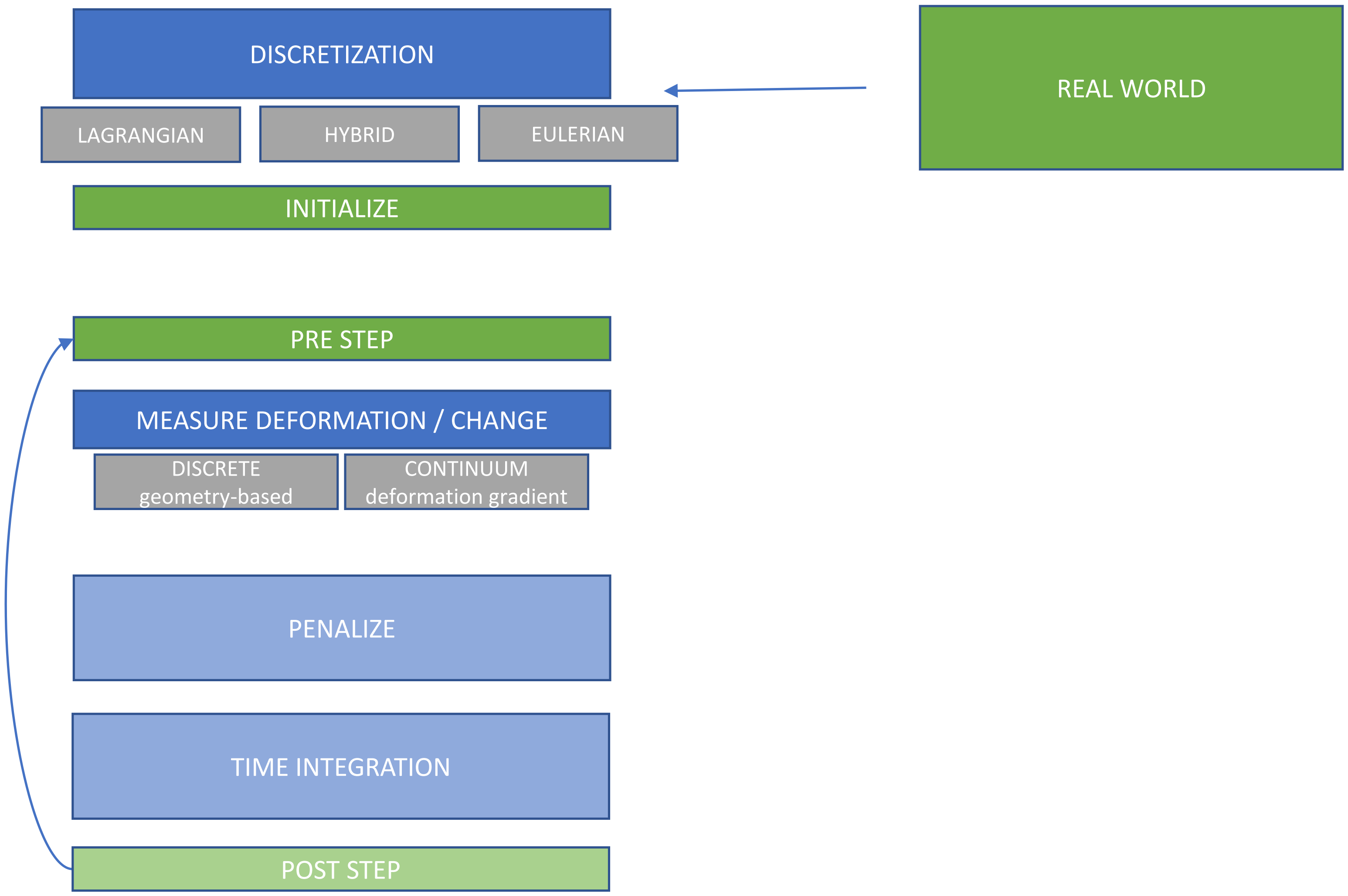


$$\boxed{F = \begin{vmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{vmatrix}}$$

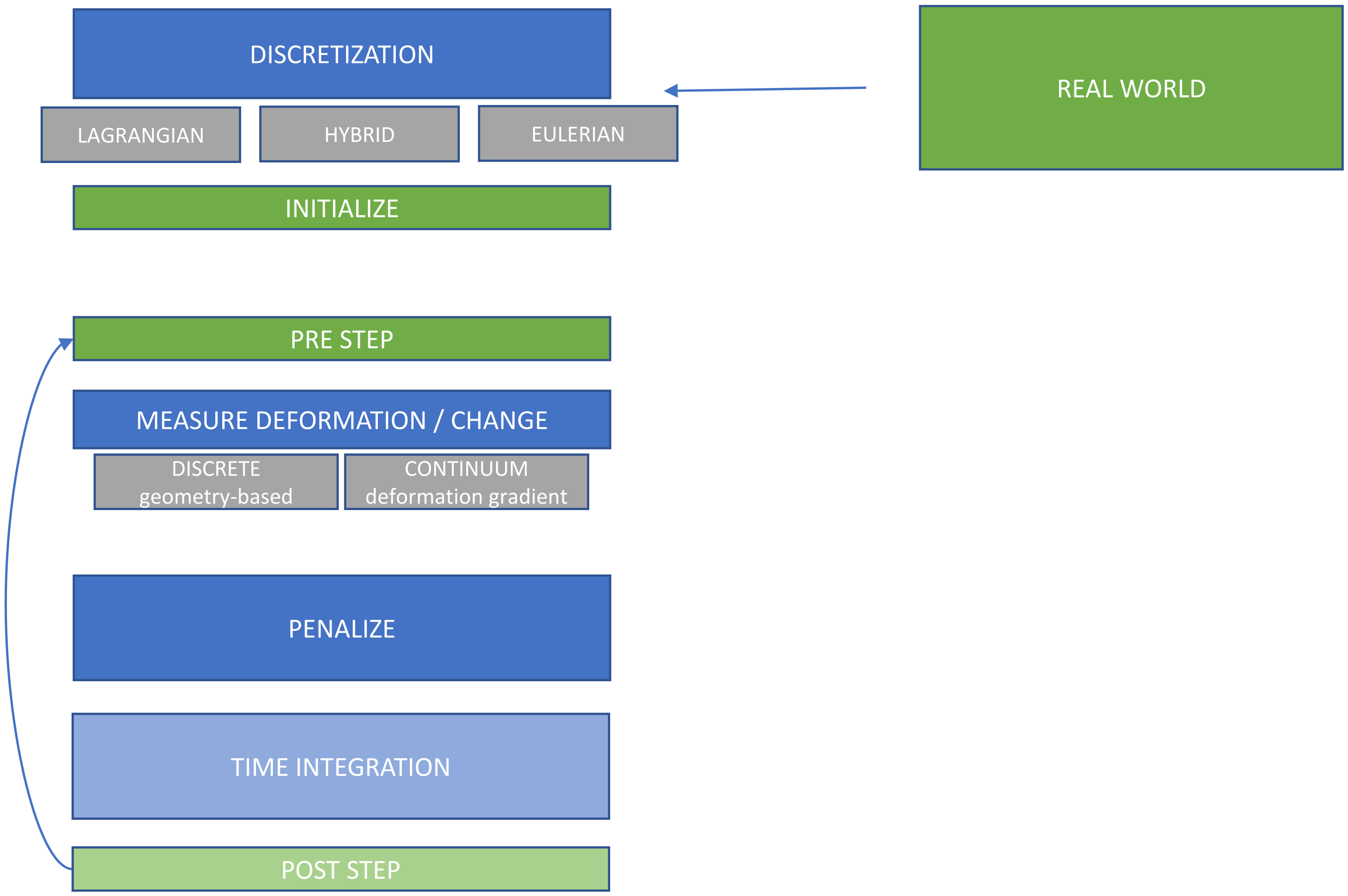
WHAT DO SOLVERS DO



WHAT DO SOLVERS DO



WHAT DO SOLVERS DO



PENALIZE

PENALIZE

- *FORCE BASED*
- *VELOCITY BASED (IMPULSE)*
- *POSITION BASED*

PENALIZE

- $\boxed{\textit{FORCE BASED}}$
- *VELOCITY BASED (IMPULSE)*
- *POSITION BASED*

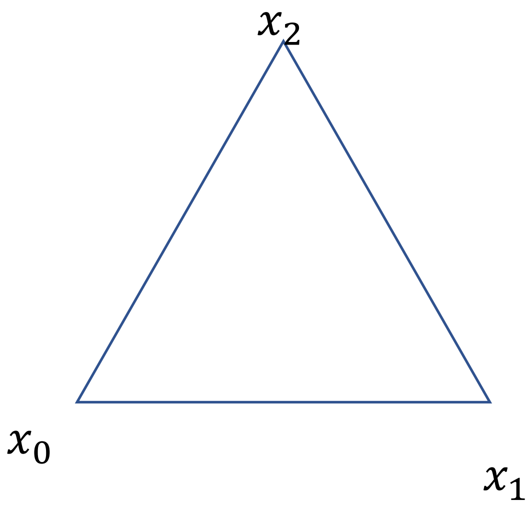
ENERGY FROM DISCRETE MODEL

GEOMETRIC DEFORMATION

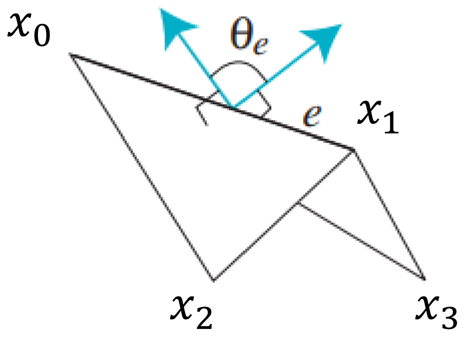
- Geometric Deformation => Energy (positive scalar function)

Distance Constraint

Area Constraint



Bending Constraint



ENERGY FROM DISCRETE MODEL

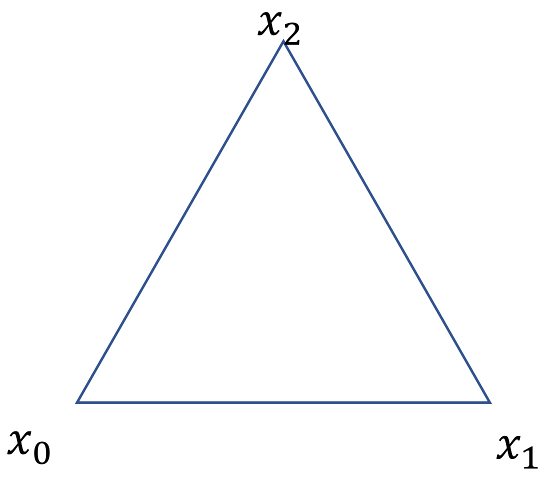
GEOMETRIC DEFORMATION

- Geometric Deformation => Energy (positive scalar function)

Distance Constraint

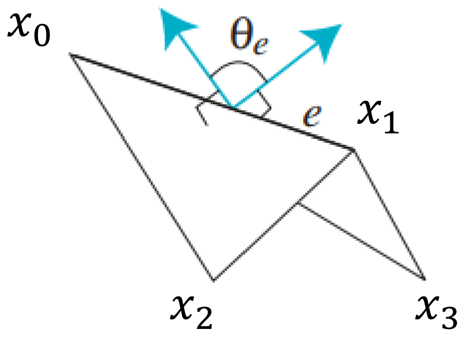
$C(x0, x1) = (1 - \|e\|/\|\bar{e}\|)^2 \|\bar{e}\|$

Area Constraint



$C(x0, x1, x2) = (1 - \|A\|/\|\bar{A}\|)^2 \|\bar{A}\|$

Bending Constraint



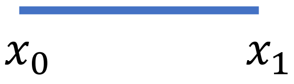
$C(x0, x1, x2, x3) = (\theta_e - \bar{\theta}_e)^2 \|\bar{e}\| / \bar{h}_e$

ENERGY FROM DISCRETE MODEL

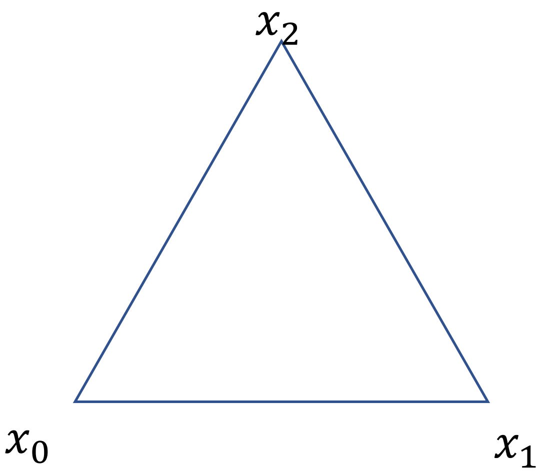
GEOMETRIC DEFORMATION

- Geometric Deformation => Energy (positive scalar function)

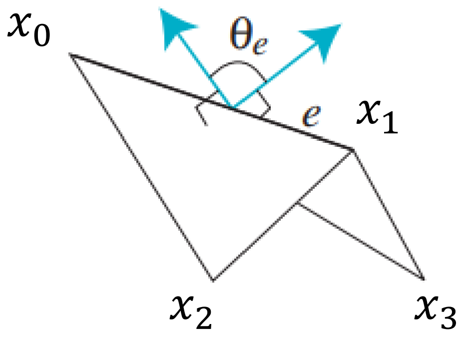
Distance Constraint



Area Constraint



Bending Constraint



C(x0, x1) = $(1-\|e\|/\|\bar{e}\|)^2\|\bar{e}\|$

C(x0, x1, x2) = $(1-\|A\|/\|\bar{A}\|)^2\|\bar{A}\|$

C(x0, x1, x2, x3) = $(\theta_e-\bar{\theta}_e)^2\|\bar{e}\|/\bar{h}_e$

$$\boxed{W(X) = C(X)\,.\,k}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model → Energy

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model → Energy

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model → Energy

Deformation Gradient

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

Strain

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

Constitutive Model

- Saint-Venant Kirchhoff
- Corotated Linear Elasticity
- Neo-Hookean
- …

Energy

User Parameters
- *Poisson ratio (v)*
- *Young's modulus (Y)*

→

Lamé constants (μ, λ)

$$\lambda = \frac{vY}{(1+v)(1-2v)}$$

$$\mu = \frac{E}{2(1+v)}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model (Strain Energy Density Function (ψ)) → Energy

Deformation Gradient

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

Strain

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

Constitutive Model

Strain Energy Density Function (ψ)

- **Saint-Venant Kirchhoff**
- Corotated Linear Elasticity
- Neo-Hookean
- …

$$\Psi(\mathbf{F}) = \mu \mathbf{E} : \mathbf{E} + \frac{\lambda}{2}\mathrm{tr}^2(\mathbf{E})$$

Energy

User Parameters

- *Poisson ratio (v)*
- *Young's modulus (Y)*

→

Lamé constants (μ, λ)

$$\lambda = \frac{vY}{(1+v)(1-2v)}$$

$$\mu = \frac{E}{2(1+v)}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model (Strain Energy Density Function (ψ)) → Energy

Deformation Gradient

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

Strain

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

Constitutive Model

Strain Energy Density Function (ψ)

- Saint-Venant Kirchhoff
- **Corotated Linear Elasticity**
- Neo-Hookean
- …

$$\Psi(\mathbf{F}) = \mu \|\mathbf{S} - \mathbf{I}\|_F^2 + (\lambda/2)\mathrm{tr}^2(\mathbf{S} - \mathbf{I})$$

$$\mathbf{F} = \mathbf{R}\mathbf{S}$$

Energy

User Parameters

- *Poisson ratio (v)*
- *Young's modulus (Y)*

→ Lamé constants (μ, λ)

$$\lambda = \frac{vY}{(1+v)(1-2v)}$$

$$\mu = \frac{E}{2(1+v)}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model (Strain Energy Density Function (ψ)) → Energy

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

- Saint-Venant Kirchhoff
- Corotated Linear Elasticity
- **Neo-Hookean**
- …

$$\Psi(I_1, I_3) = \frac{\mu}{2}(I_1 - \log(I_3) - 3) + \frac{\lambda}{8}\log^2(I_3)$$

$$I_1 = \mathrm{tr}(\mathbf{F}^T\mathbf{F})$$

$$I_3 = \det(\mathbf{F}^T\mathbf{F})$$

User Parameters: *Poisson ratio (v)*, *Young's modulus (Y)* →

Lamé constants (μ, λ)

$$\lambda = \frac{vY}{(1+v)(1-2v)}$$

$$\mu = \frac{E}{2(1+v)}$$

ENERGY FROM CONTINOUS MODEL

CONTINUUM MECHANICS – DEFORMATION TO ENERGY

Deformation Gradient → Strain → Constitutive Model (Strain Energy Density Function (ψ)) → scale by volume → Energy

Deformation Gradient

$$F = \mathrm{I} + \frac{\partial u}{\partial X}$$

Strain

- Linear Cauchy

$$E = \frac{1}{2}(F^T + F) - I$$

- Non-linear Green

$$E = \frac{1}{2}(F^T F - I)$$

Constitutive Model

Strain Energy Density Function (ψ)

- Saint-Venant Kirchhoff
- Corotated Linear Elasticity
- Neo-Hookean
- …

Energy

W(X)

User Parameters

- *Poisson ratio (v)*
- *Young's modulus (Y)*

→

Lamé constants (μ, λ)

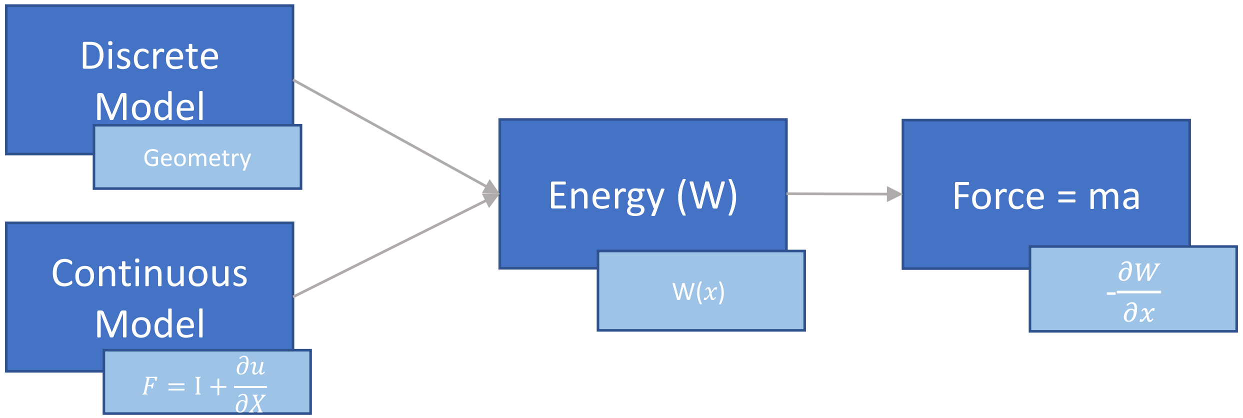
$$\lambda = \frac{vY}{(1+v)(1-2v)}$$

$$\mu = \frac{E}{2(1+v)}$$

FROM ENERGY TO FORCE

DEFINITIONS

- A force is the negative of the derivate(slope) of the potential energy



FROM ENERGY TO FORCE

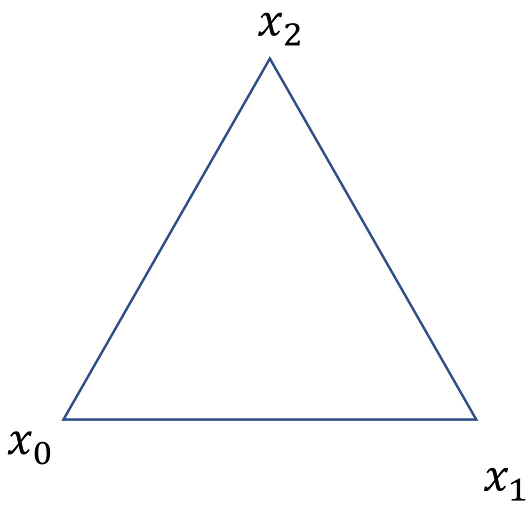
ILLUSTRATIONS

Distance Constraint
To Forces

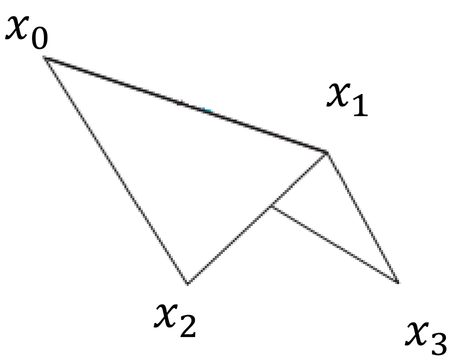
$f0 = -\frac{\partial W}{\partial x0}$ $\quad$ $f1 = -\frac{\partial W}{\partial x1}$

$f0 + f1 = (0,0)$

Area Constraint
to Forces



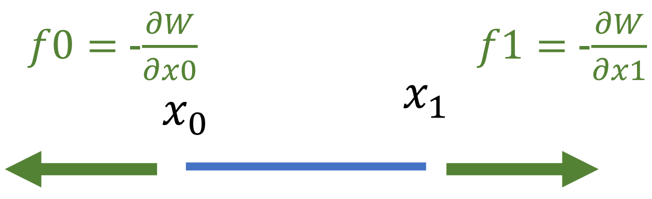
Bending Constraint
to Forces



FROM ENERGY TO FORCE

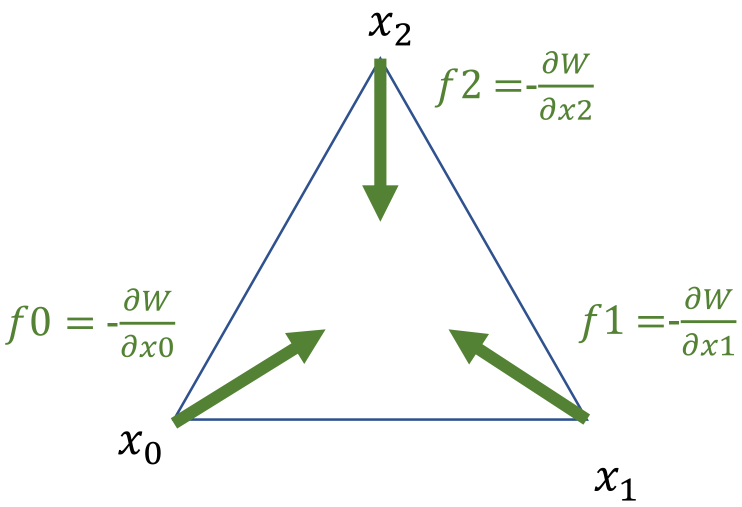
ILLUSTRATIONS

Distance Constraint
To Forces



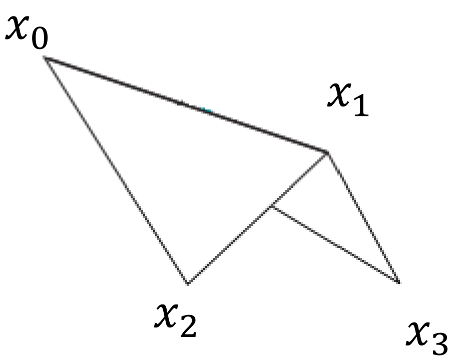
$f0 + f1 = (0,0)$

Area Constraint
to Forces



$f0 + f1 + f2 = (0,0)$

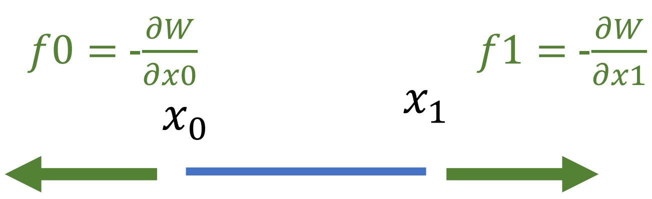
Bending Constraint
to Forces



FROM ENERGY TO FORCE

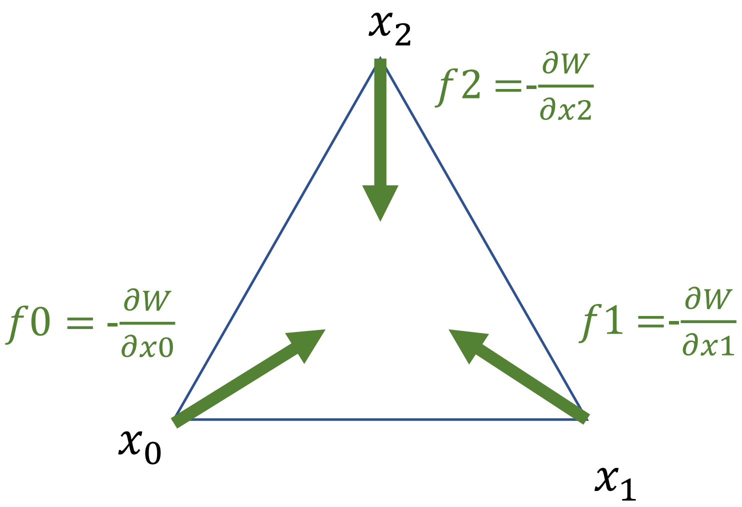
ILLUSTRATIONS

Distance Constraint
To Forces



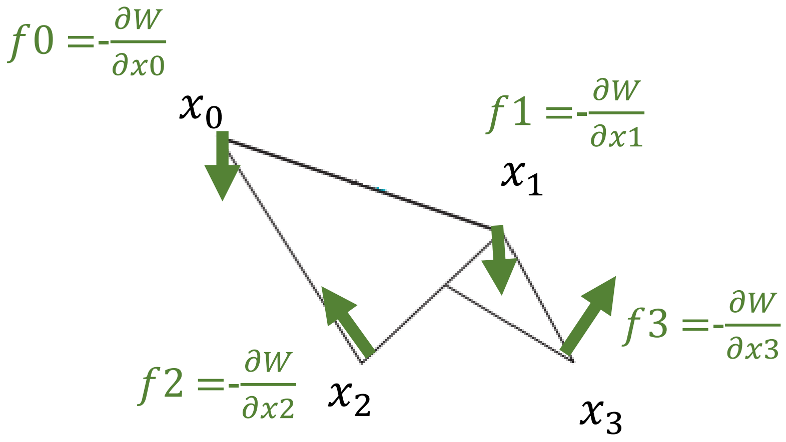
$f0 + f1 = (0,0)$

Area Constraint
to Forces



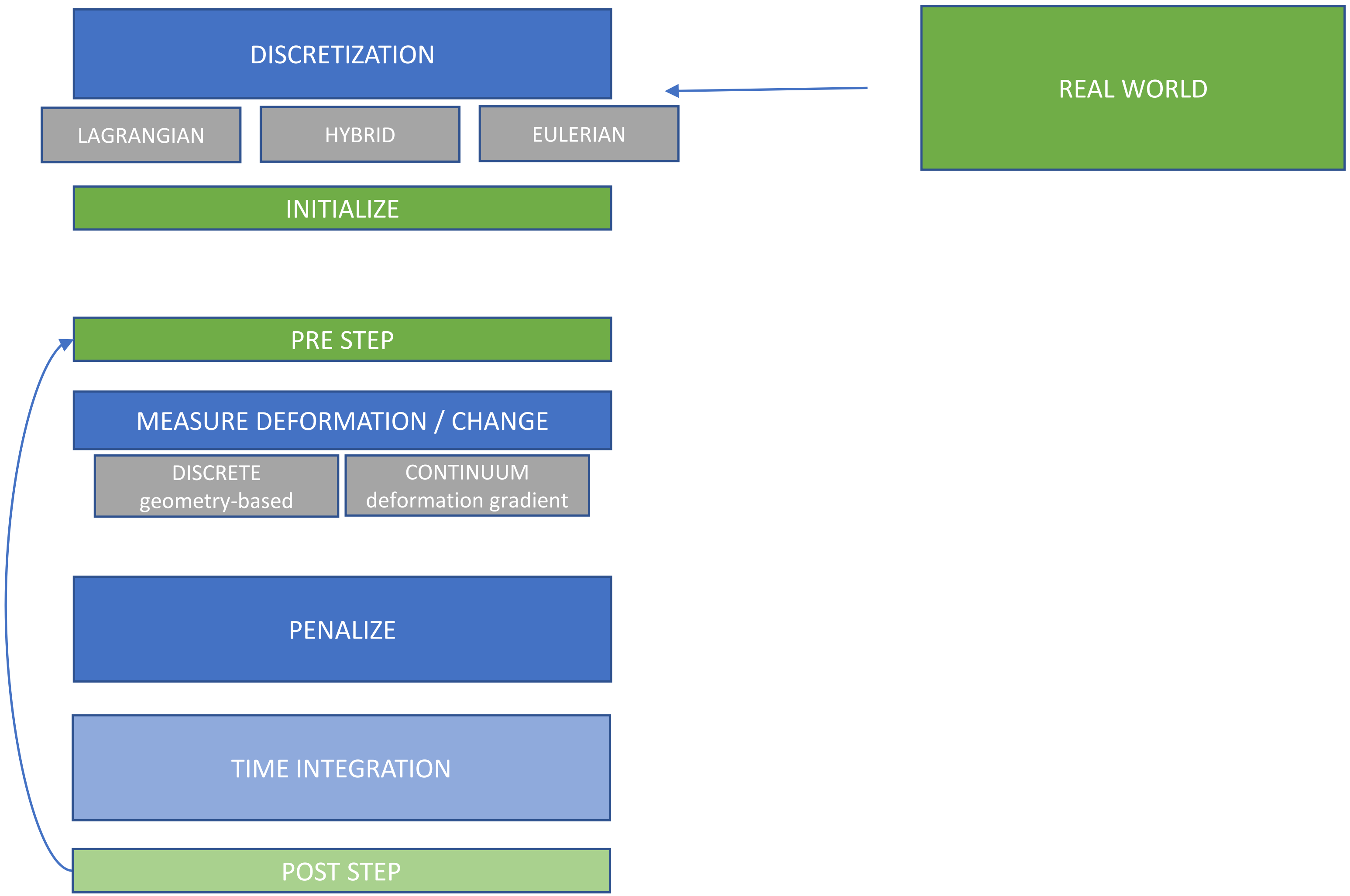
$f0 + f1 + f2 = (0,0)$

Bending Constraint
to Forces

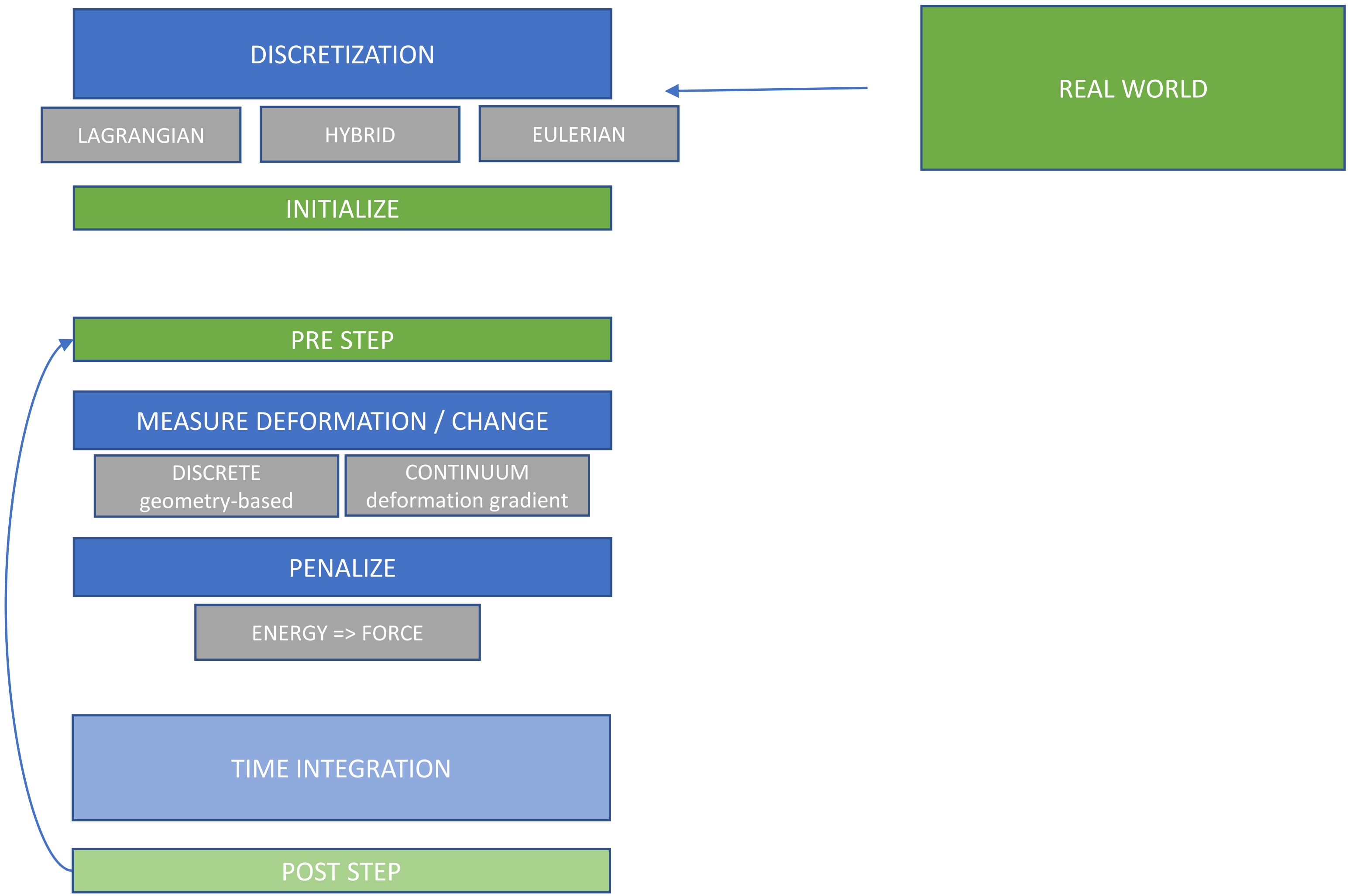


$f0 + f1 + f2 + f3 = (0,0,0)$

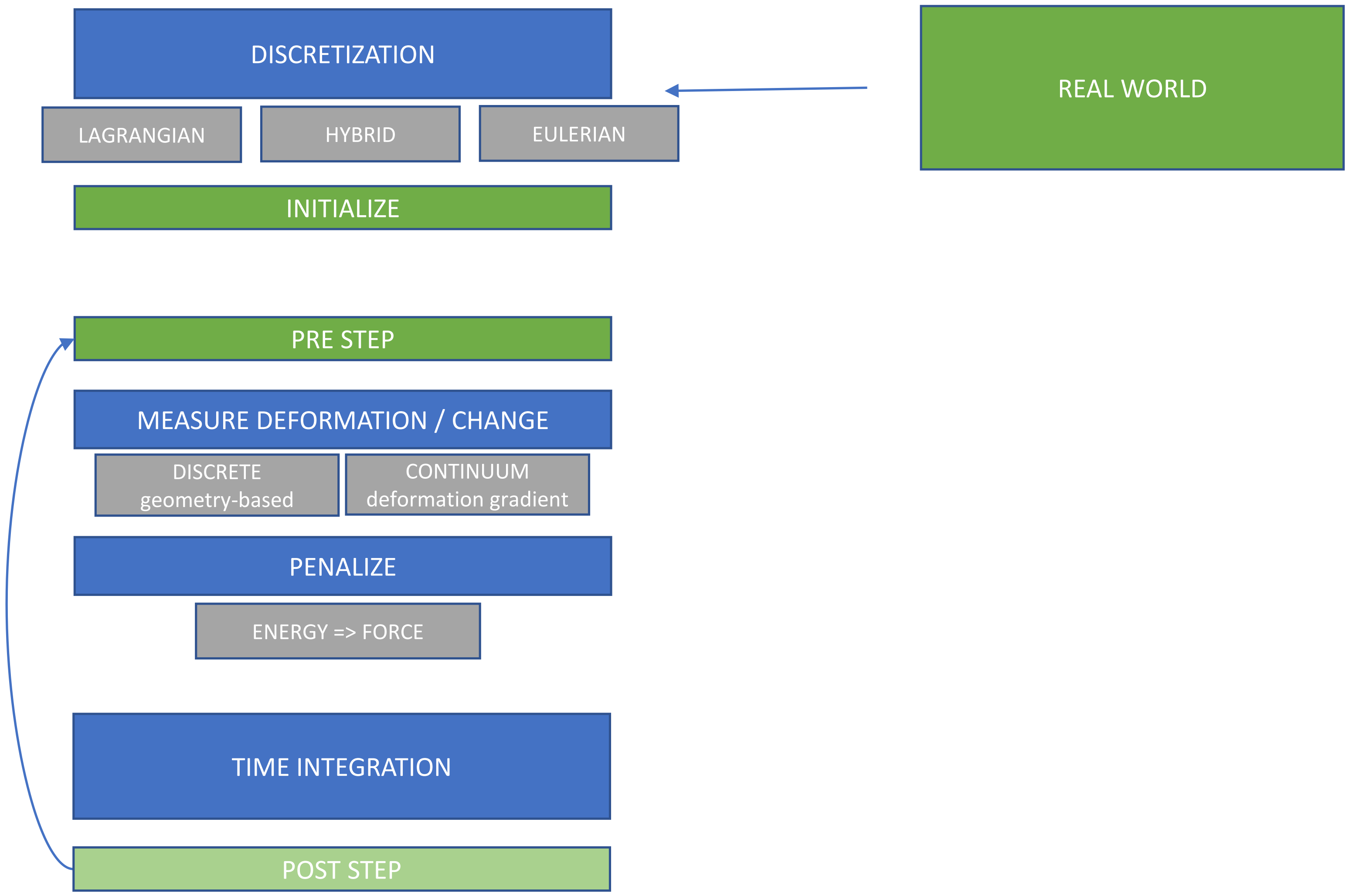
WHAT DO SOLVERS DO



WHAT DO SOLVERS DO

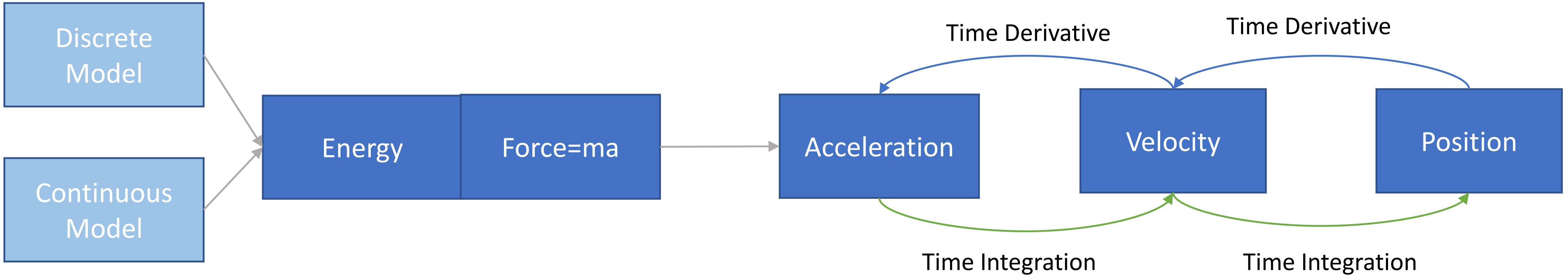


WHAT DO SOLVERS DO



TIME INTEGRATION

PARTICLE STATE



EXPLICIT VS IMPLICIT

TIME INTEGRATION

EXPLICIT INTEGRATOR

- *Forward Euler (Explicit Euler)*
- *Runge Kutta (RK2, RK4, RK…)*
- *Mid point*

- Easy to implement
- Conditionally stable

IMPLICIT INTEGRATORS

- *Backward Euler (Implicit Euler)*
- *Higher order methods*

- Difficult to implement
- Unconditionally stable

EXPLICIT VS IMPLICIT

TIME INTEGRATION

h : timestep (float)

m: mass (float)

v : current velocity(vector2)

p : current position(vector2)

$f(\quad)$: force function (vector2)

EXPLICIT VS IMPLICIT

TIME INTEGRATION

h : timestep (float)

m: mass (float)

v : current velocity(vector2)

p : current position(vector2)

$f(\quad)$: force function (vector2)

EXPLICIT INTEGRATOR

$$\text{next\_v} = \text{v} + h\frac{f(p)}{m}$$

$$\text{next\_p} = \text{p} + h\text{v}$$

EXPLICIT VS IMPLICIT

TIME INTEGRATION

h : timestep (float)

m: mass (float)

v : current velocity(vector2)

p : current position(vector2)

$f(\quad)$: force function (vector2)

EXPLICIT INTEGRATOR

$$\text{next\_v} = \text{v} + h\frac{f(p)}{m}$$

$$\text{next\_p} = \text{p} + h\text{v}$$

IMPLICIT INTEGRATOR

$$\text{next\_v} = \text{v} + h\frac{f(next\_p)}{m}$$

$$\text{next\_p} = \text{p} + h\ \text{next\_v}$$

EXPLICIT VS IMPLICIT

TIME INTEGRATION

h : timestep (float)

m: mass (float)

v : current velocity(vector2)

p : current position(vector2)

$f(\quad)$: force function (vector2)

EXPLICIT INTEGRATOR

- Single line
- Conditionally stable

$$\text{next\_v} = \text{v} + h\frac{f(p)}{m}$$

$$\text{next\_p} = \text{p} + h\text{v}$$

IMPLICIT INTEGRATOR

- Solve sparse system
- Unconditionally stable

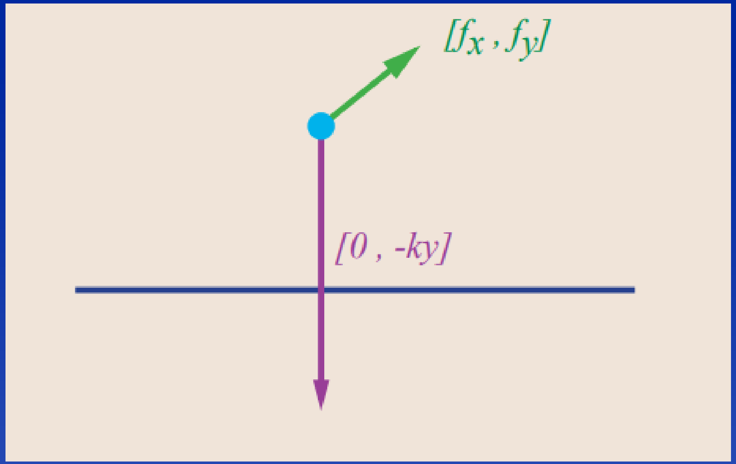
$$\text{next\_v} = \text{v} + h\frac{f(next\_p)}{m}$$

$$\text{next\_p} = \text{p} + h\ \text{next\_v}$$

STABILITY EXAMPLE

Example: particle-on-line

- A particle P in the plane.
- Interactive "dragging" force $[f_x, f_y]$.
- A penalty force $[0,-ky]$ tries to keep P on the x-axis.

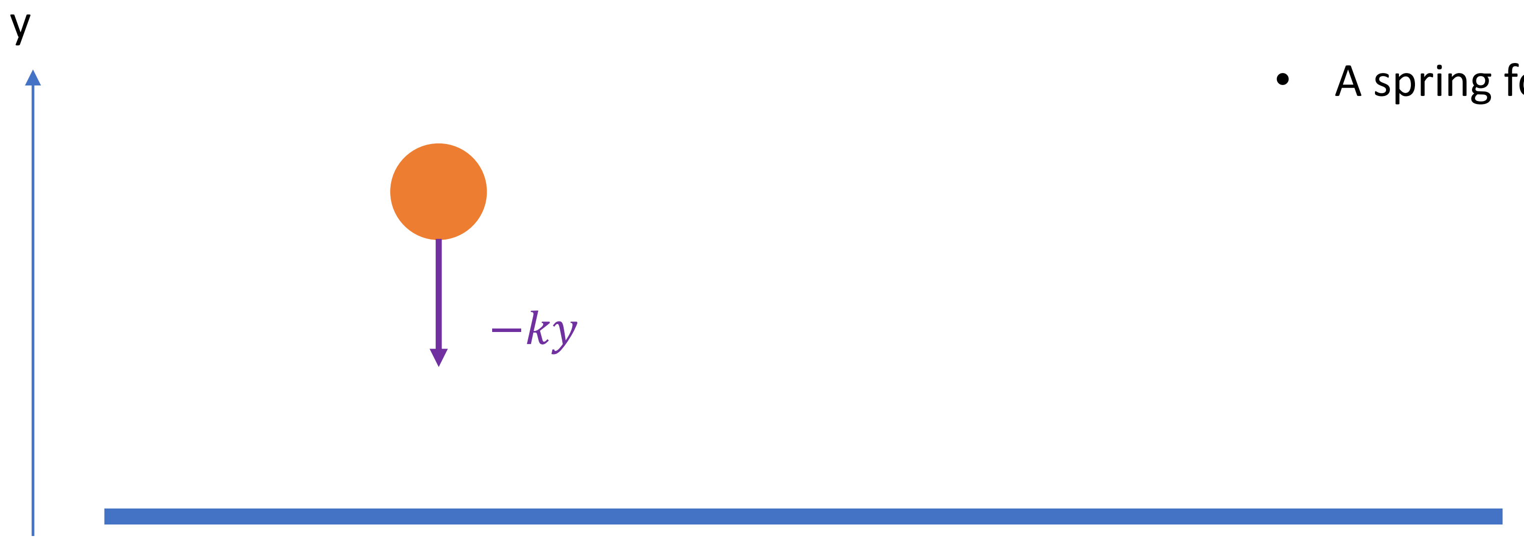


- Suppose you want P to stay within a miniscule ε of the x-axis when you try to pull it off with a huge force f_{max}.
- How big does k have to be? How *small* must h be?

Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blowup." ACM Transactions onGraphics (SIGGRAPH 1997) (1997).

STABILITY EXAMPLE

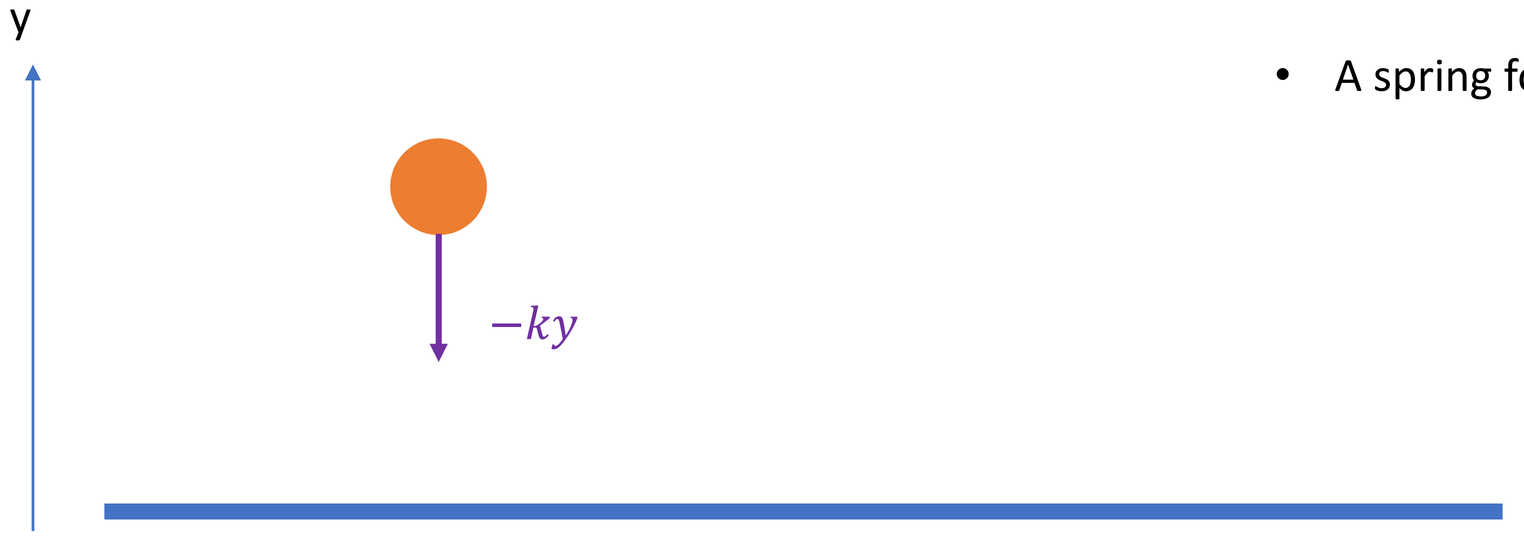
TIME INTEGRATION



- A spring force $-ky$ tries to keep the particle on the blue line

STABILITY EXAMPLE

TIME INTEGRATION



- A spring force -ky tries to keep the particle on the blue line

EXPLICIT INTEGRATOR

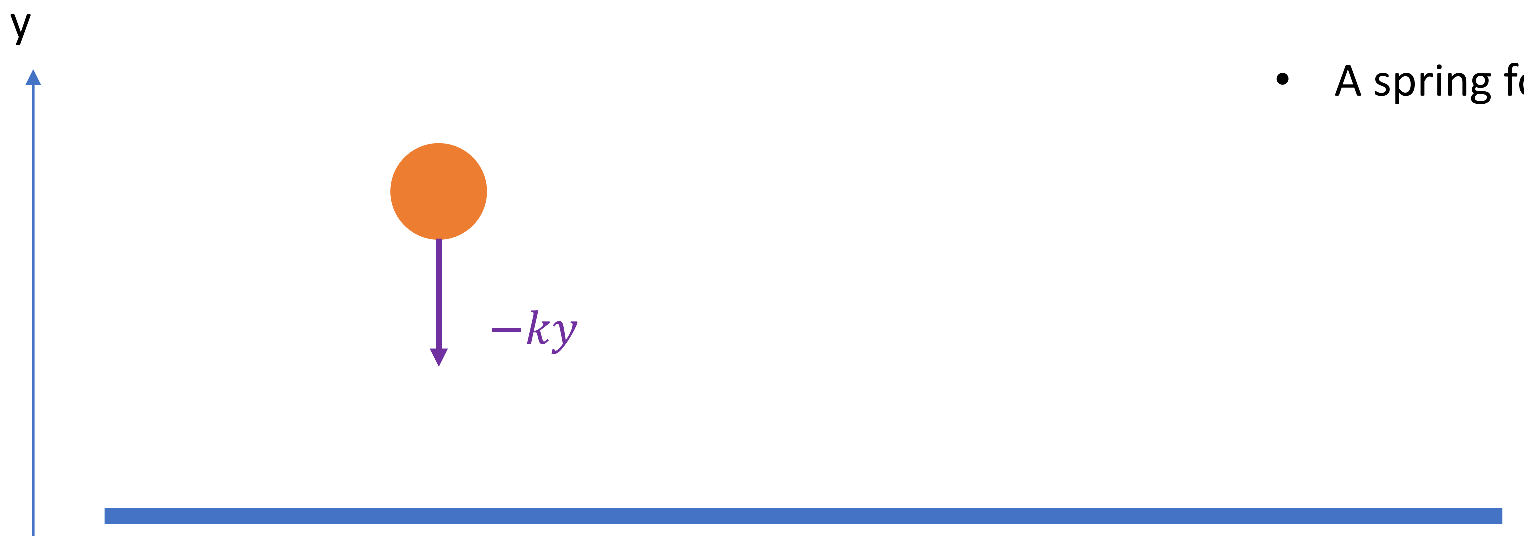
$$\mathrm{next\_y} = \mathrm{y} + h\frac{-ky}{m}$$

IMPLICIT INTEGRATOR

$$\mathrm{next\_y} = \mathrm{y} + h\frac{-k.next\_y}{m}$$

STABILITY EXAMPLE

TIME INTEGRATION



- A spring force -ky tries to keep the particle on the blue line

EXPLICIT INTEGRATOR

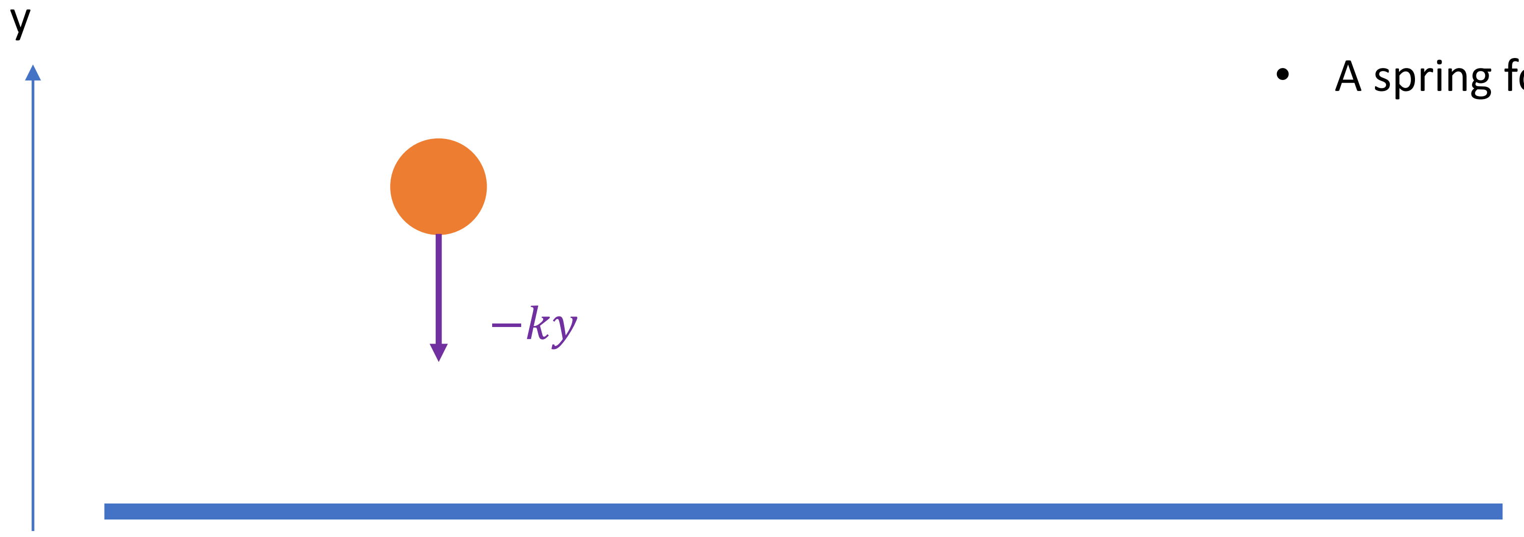
$$\boxed{\mathrm{next\_y} = \mathrm{y} + h\frac{-ky}{m}}$$

IMPLICIT INTEGRATOR

$$\mathrm{next\_y} = \mathrm{y} + h\frac{-k.next\_y}{m}$$

STABILITY EXAMPLE

TIME INTEGRATION



- A spring force -ky tries to keep the particle on the blue line

EXPLICIT INTEGRATOR

$$\boxed{\text{next\_y} = \text{y} + h\frac{-ky}{m}}$$

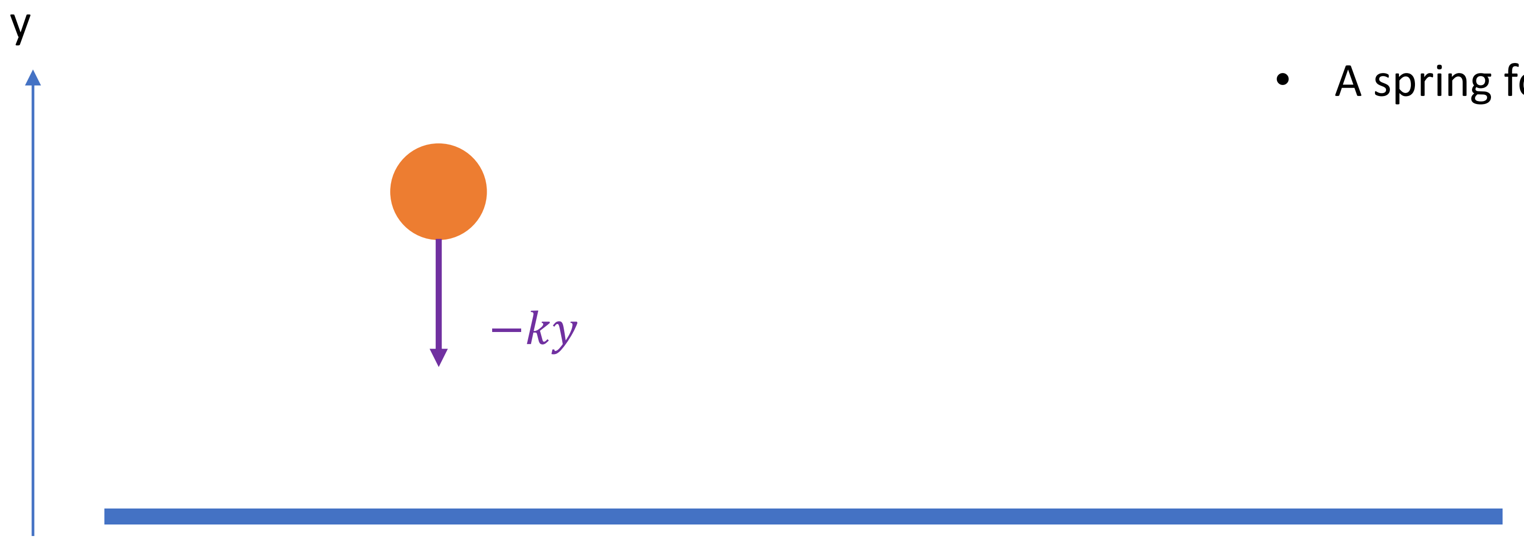
IMPLICIT INTEGRATOR

$$\text{next\_y} = \text{y} + h\frac{-k.next\_y}{m}$$

$$\text{next\_y} + h\frac{k.next\_y}{m} = \text{y}$$

STABILITY EXAMPLE

TIME INTEGRATION



- A spring force -ky tries to keep the particle on the blue line

EXPLICIT INTEGRATOR

$$\boxed{\text{next\_y} = \text{y} + h\frac{-ky}{m}}$$

IMPLICIT INTEGRATOR

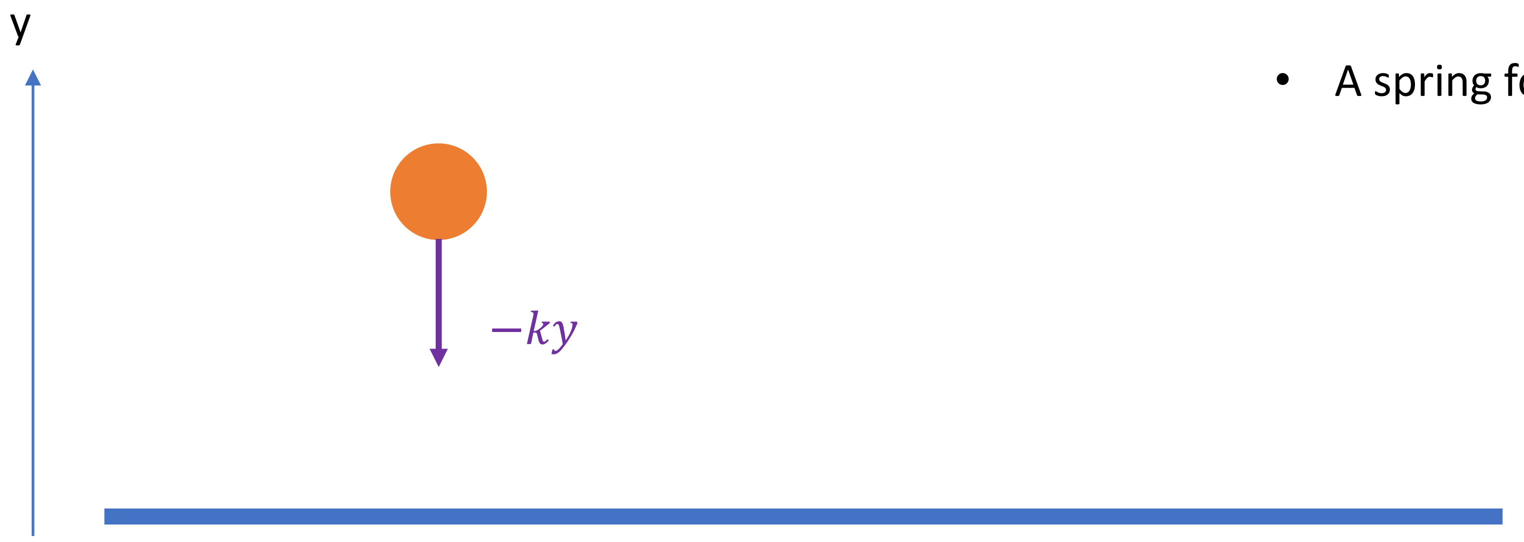
$$\text{next\_y} = \text{y} + h\frac{-k.next\_y}{m}$$

$$\text{next\_y} + h\frac{k.next\_y}{m} = \text{y}$$

$$\text{next\_y}(1 + \frac{hk}{m}) = \text{y}$$

STABILITY EXAMPLE

TIME INTEGRATION



- A spring force -ky tries to keep the particle on the blue line

EXPLICIT INTEGRATOR

$$\boxed{\text{next\_y} = \text{y} + h\frac{-ky}{m}}$$

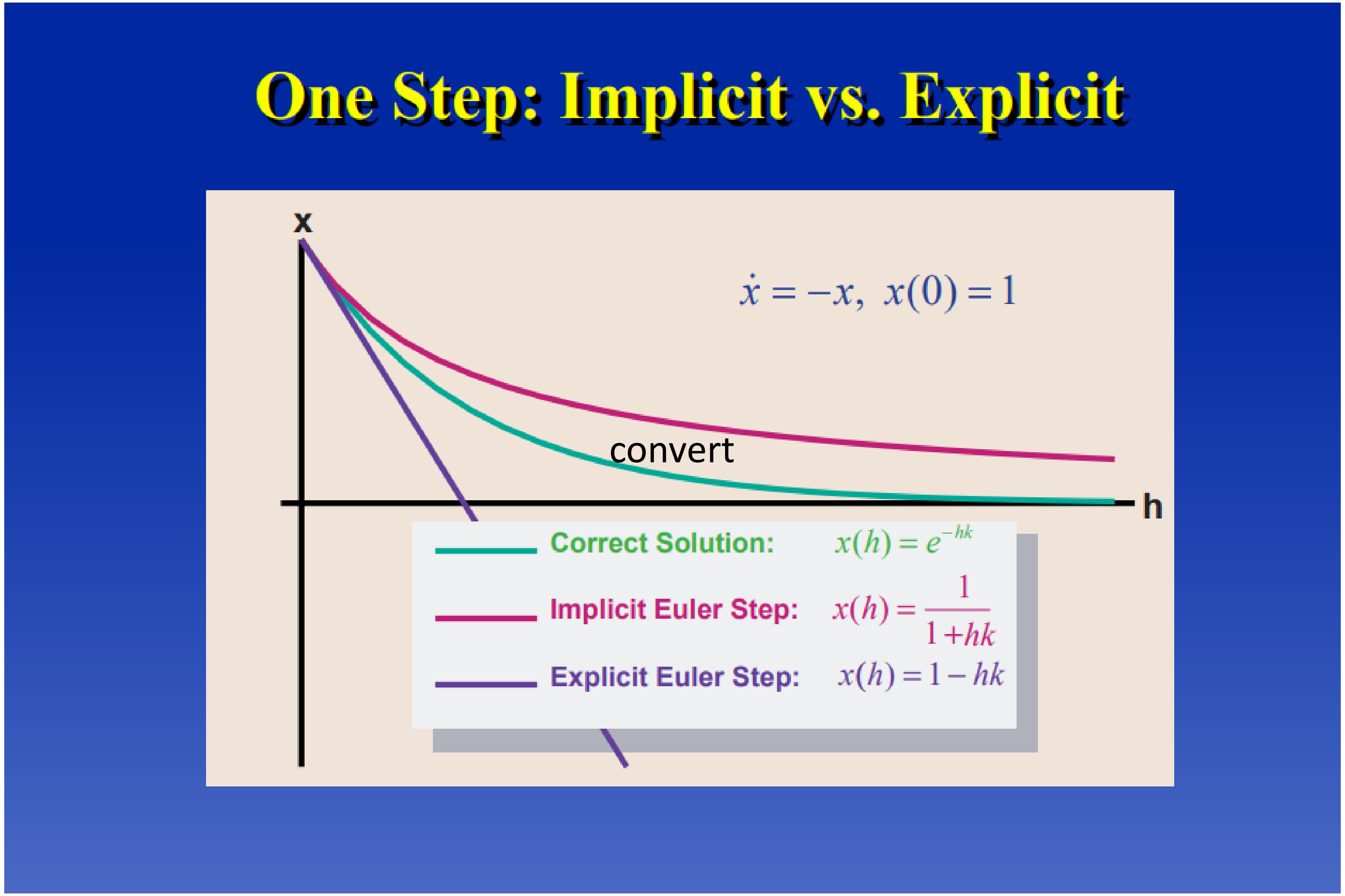
IMPLICIT INTEGRATOR

$$\text{next\_y} = \text{y} + h\frac{-k.next\_y}{m}$$

$$\text{next\_y} + h\frac{k.next\_y}{m} = \text{y}$$

$$\text{next\_y}(1 + \frac{hk}{m}) = \text{y} \Rightarrow \boxed{\text{next\_y} = \frac{\text{y}}{1 + \frac{hk}{m}}}$$

STABILITY EXAMPLE



Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blowup." ACM Transactions onGraphics (SIGGRAPH 1997) (1997).

VARIATIONAL IMPLICIT EULER

$$\mathrm{next\_v} = \mathrm{v} + h\frac{f(next\_p)}{m}$$

$$\mathrm{next\_p} = \mathrm{p} + h\ \mathrm{next\_v}$$

VARIATIONAL IMPLICIT EULER

$$\text{next\_v} = \text{v} + h\frac{f(next\_p)}{m}$$

$$\text{next\_p} = \text{p} + h\ \text{next\_v}$$

MULTIPLE PARTICLES

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

VARIATIONAL IMPLICIT EULER

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

$$x_{n+1} = x_n + h(v_n + hM^{-1}f(x_{n+1}))$$

$$x_{n+1} = x_n + hv_n + h^2M^{-1}f(x_{n+1}))$$

$$x_{n+1} - x_n - hv_n = h^2M^{-1}f(x_{n+1}))$$

$$M(x_{n+1} - x_n - hv_n) = h^2f(x_{n+1}))$$

VARIATIONAL IMPLICIT EULER

$$M(x_{n+1} - x_n - hv_n) = h^2 f(x_{n+1})$$

convert

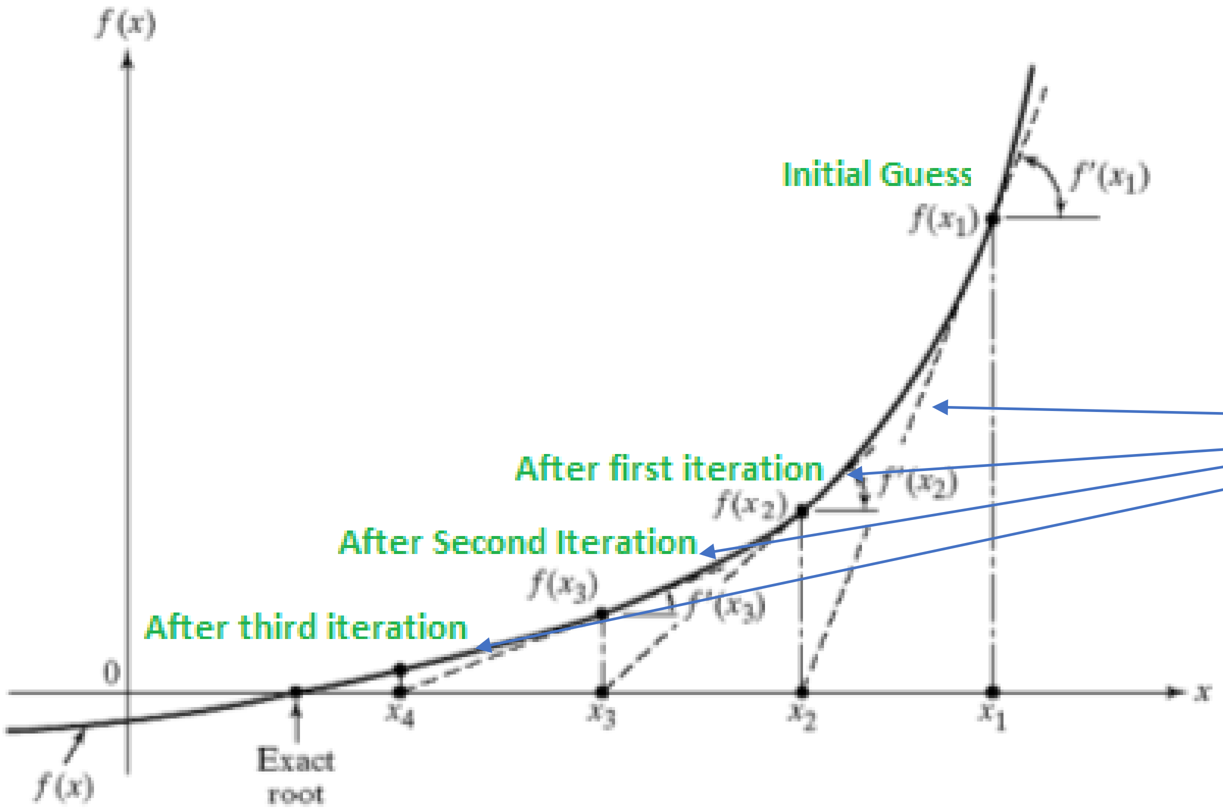
OPTIMIZATION PROBLEM

$$e(x_{n+1}) = \frac{1}{2h^2}(x_{n+1} - x_n - hv_n)^T M(x_{n+1} - x_n - hv_n) + W(x_{n+1})$$

Minimize the scalar function e(…) with argument x_{n+1}

VARIATIONAL IMPLICIT EULER

Minimize this scalar function with Newton Iterations



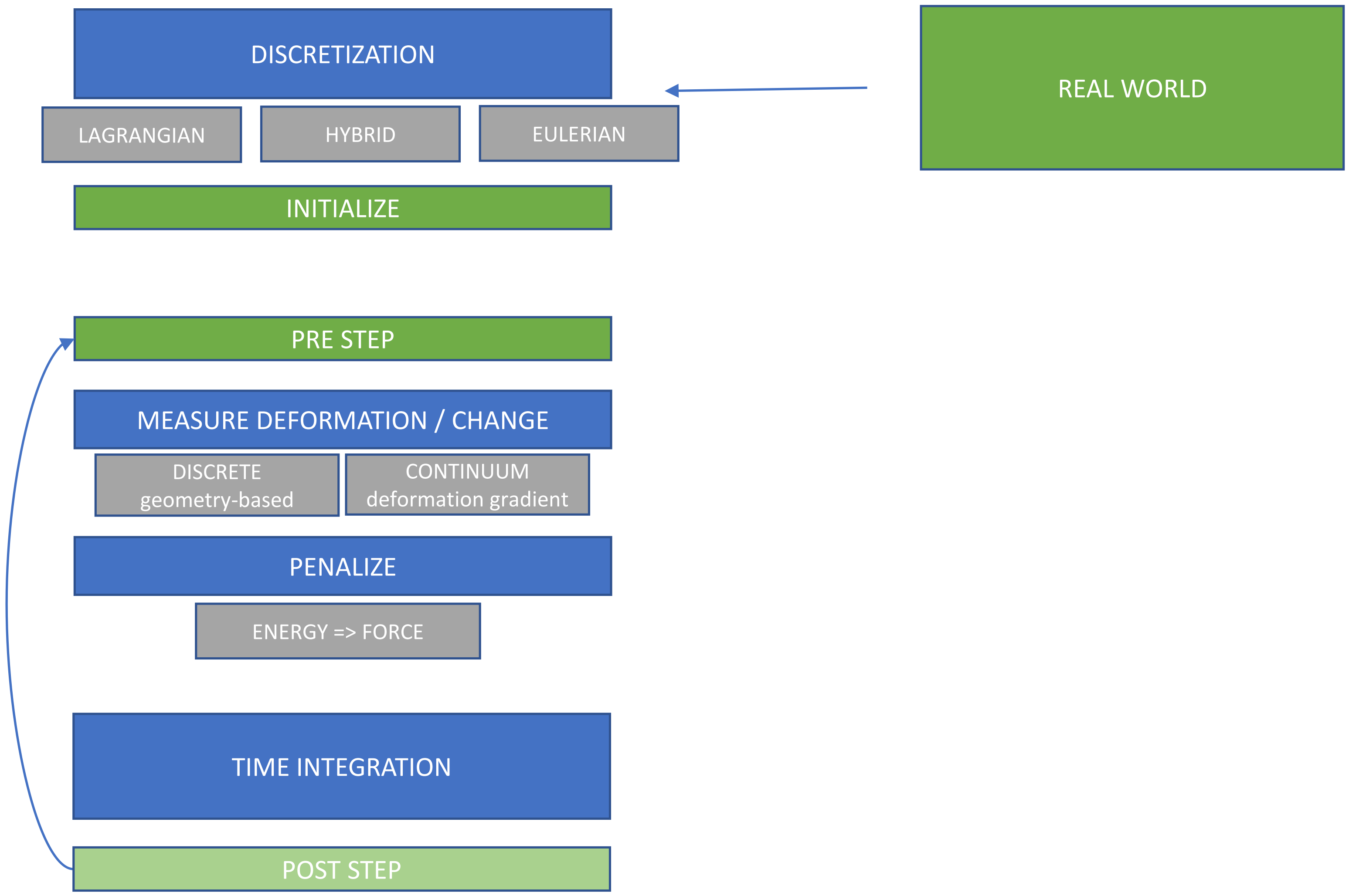
A= ∇_e^2 b $= \nabla_e$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \\ a_n \end{bmatrix}$$

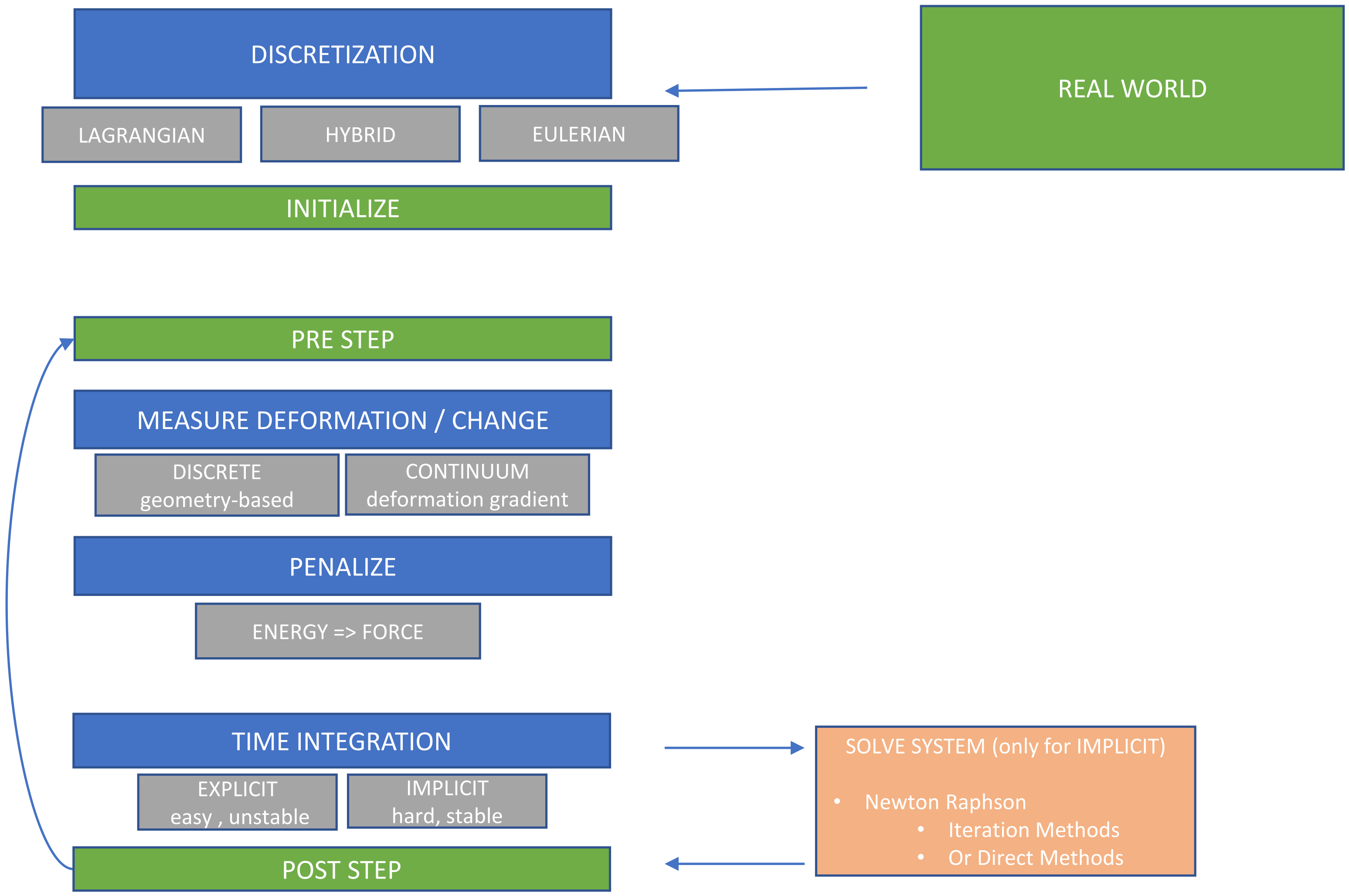
Solve Linear System

A$x_{n+1} = -$b

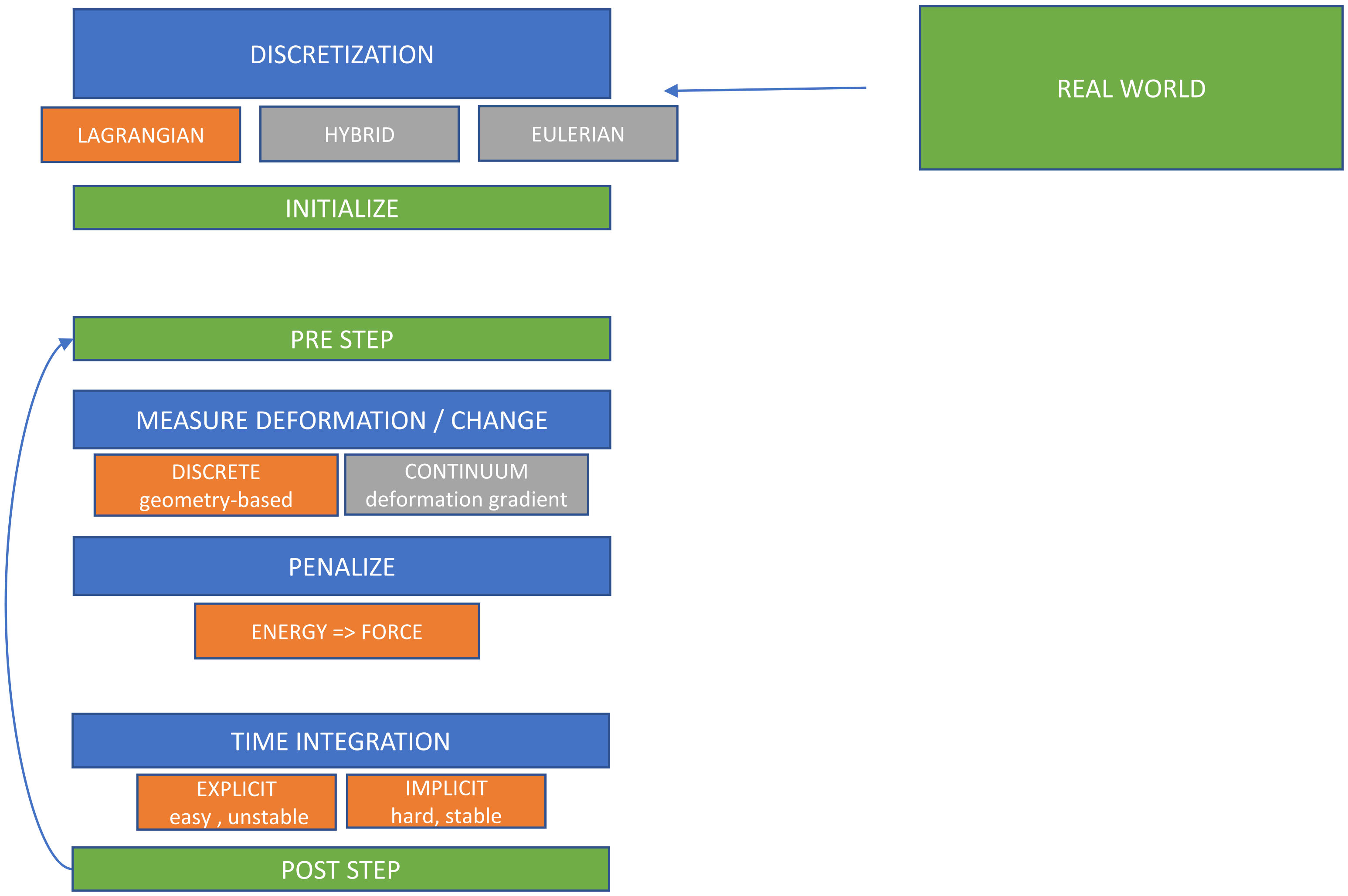
WHAT DO SOLVERS DO



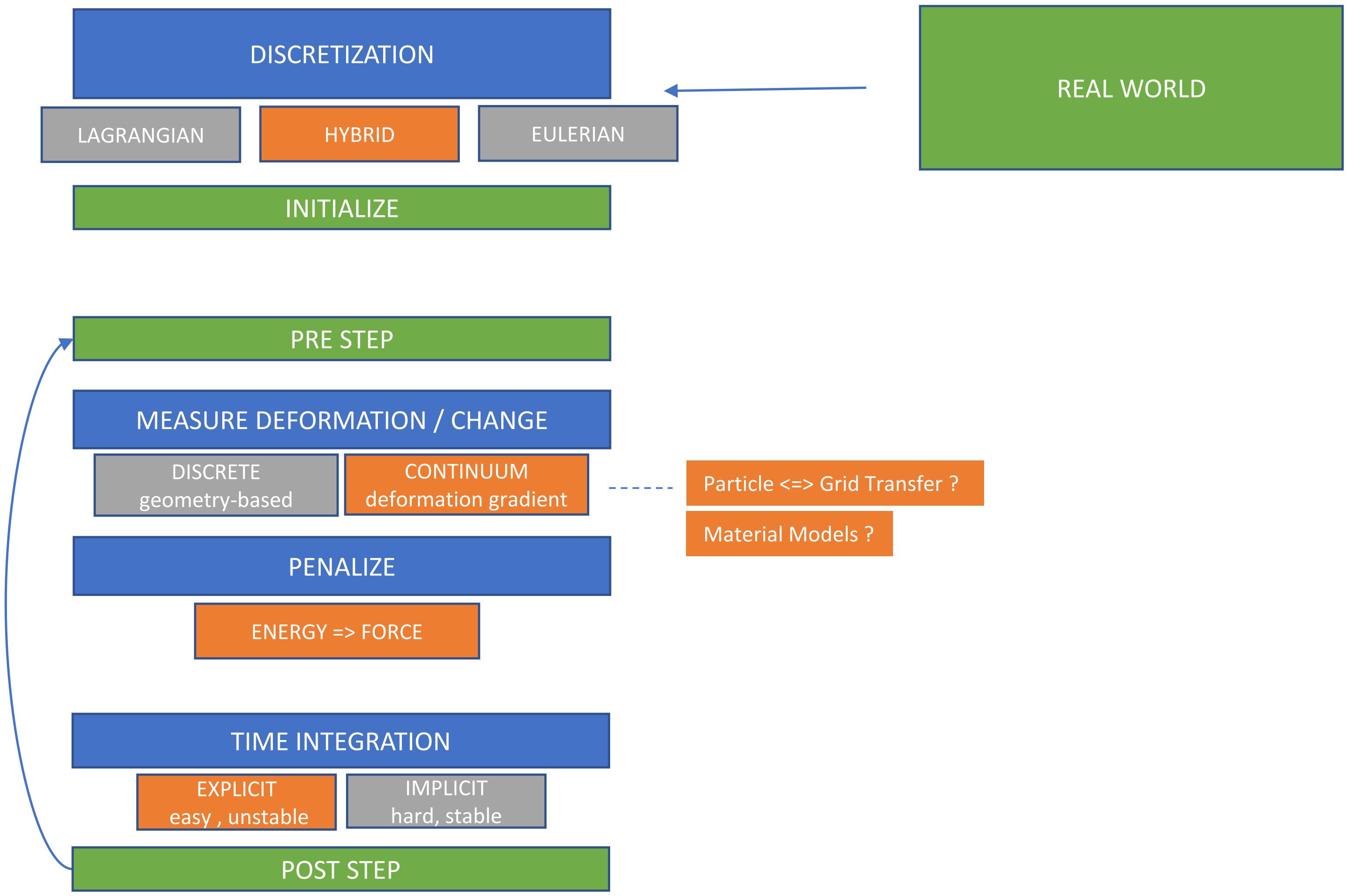
WHAT DO SOLVERS DO



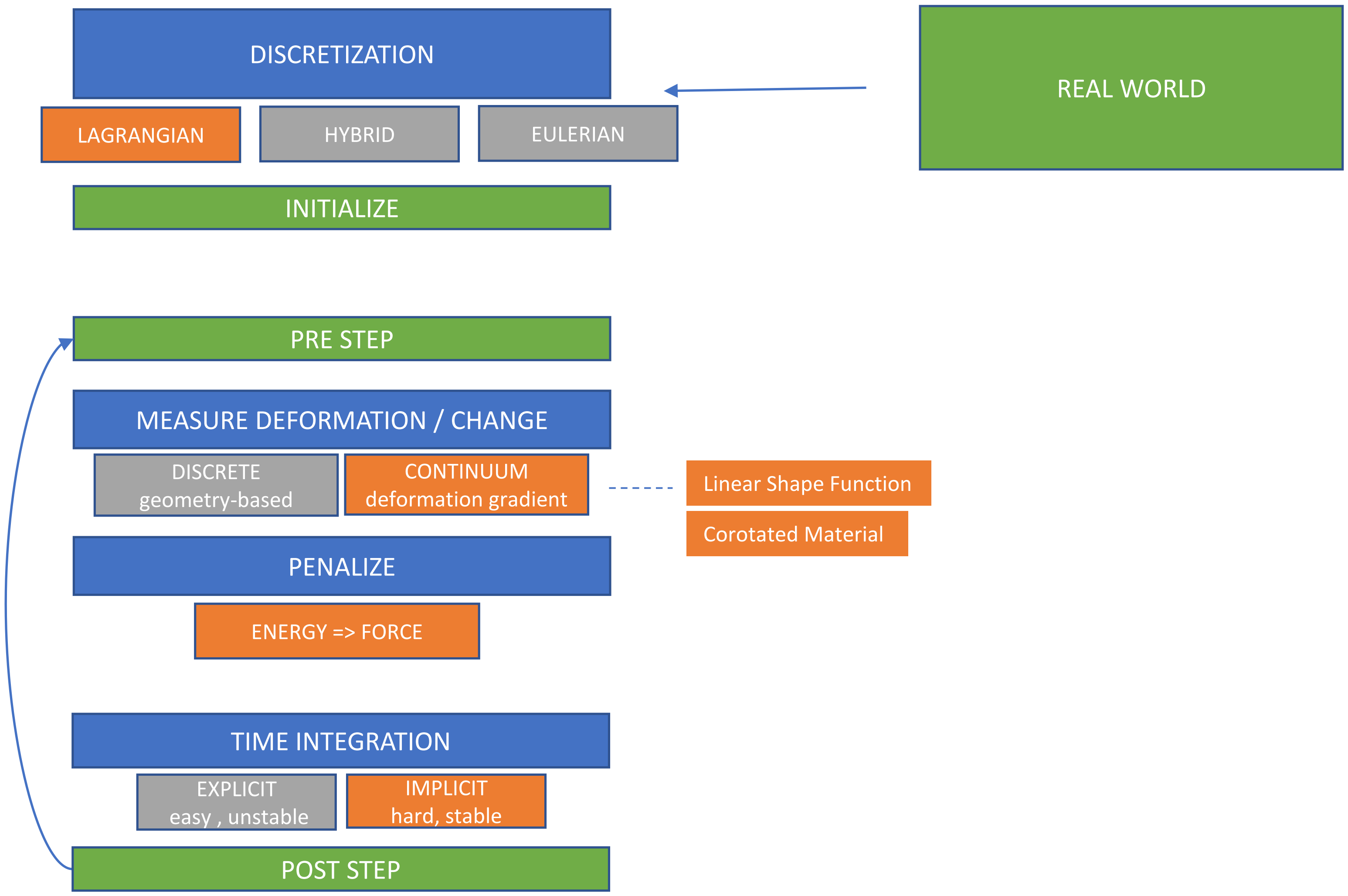
DISCRETE SHELLS



EXPLICIT MPM

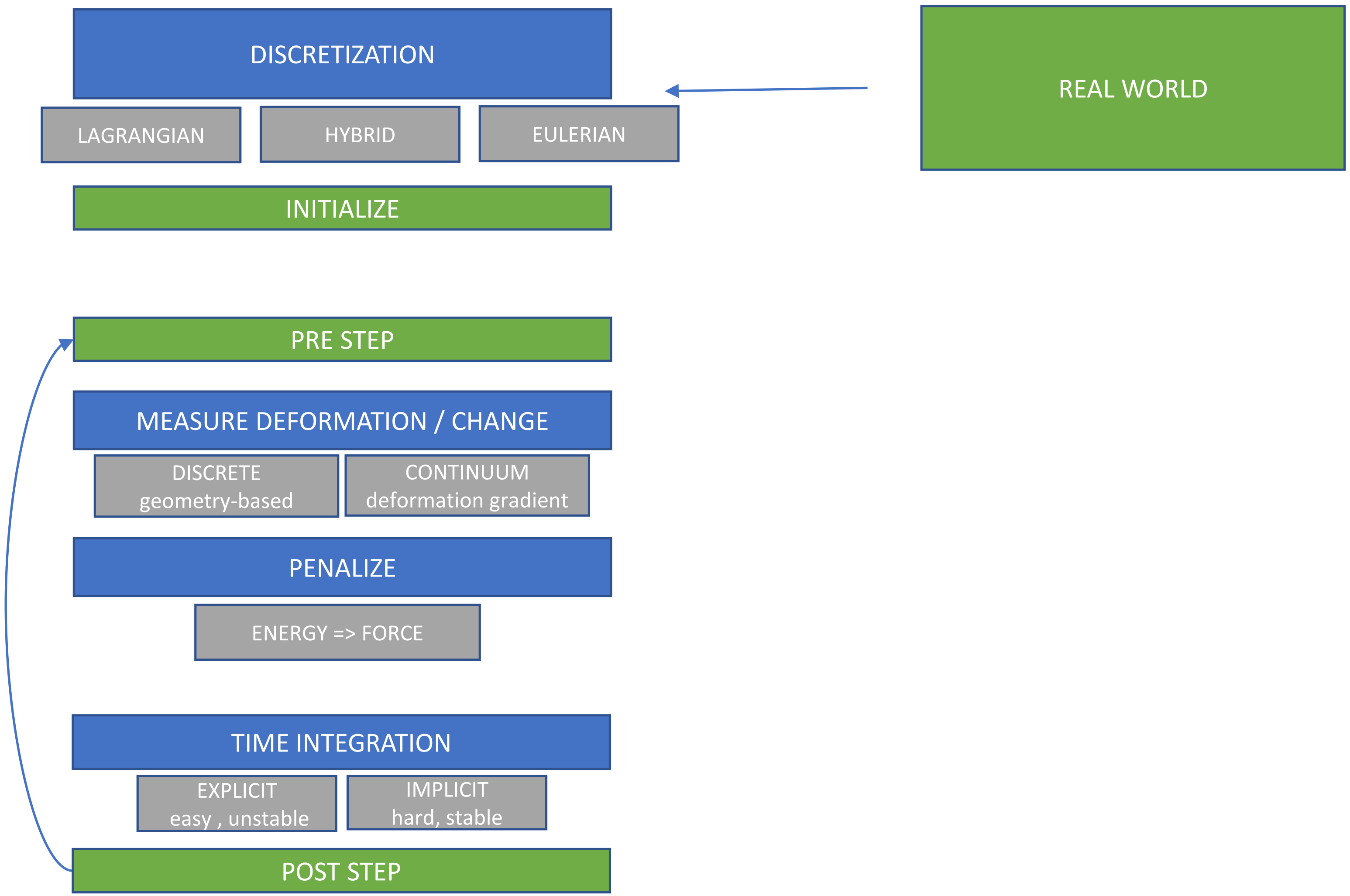


IMPLICIT COROTATED LINEAR FEM

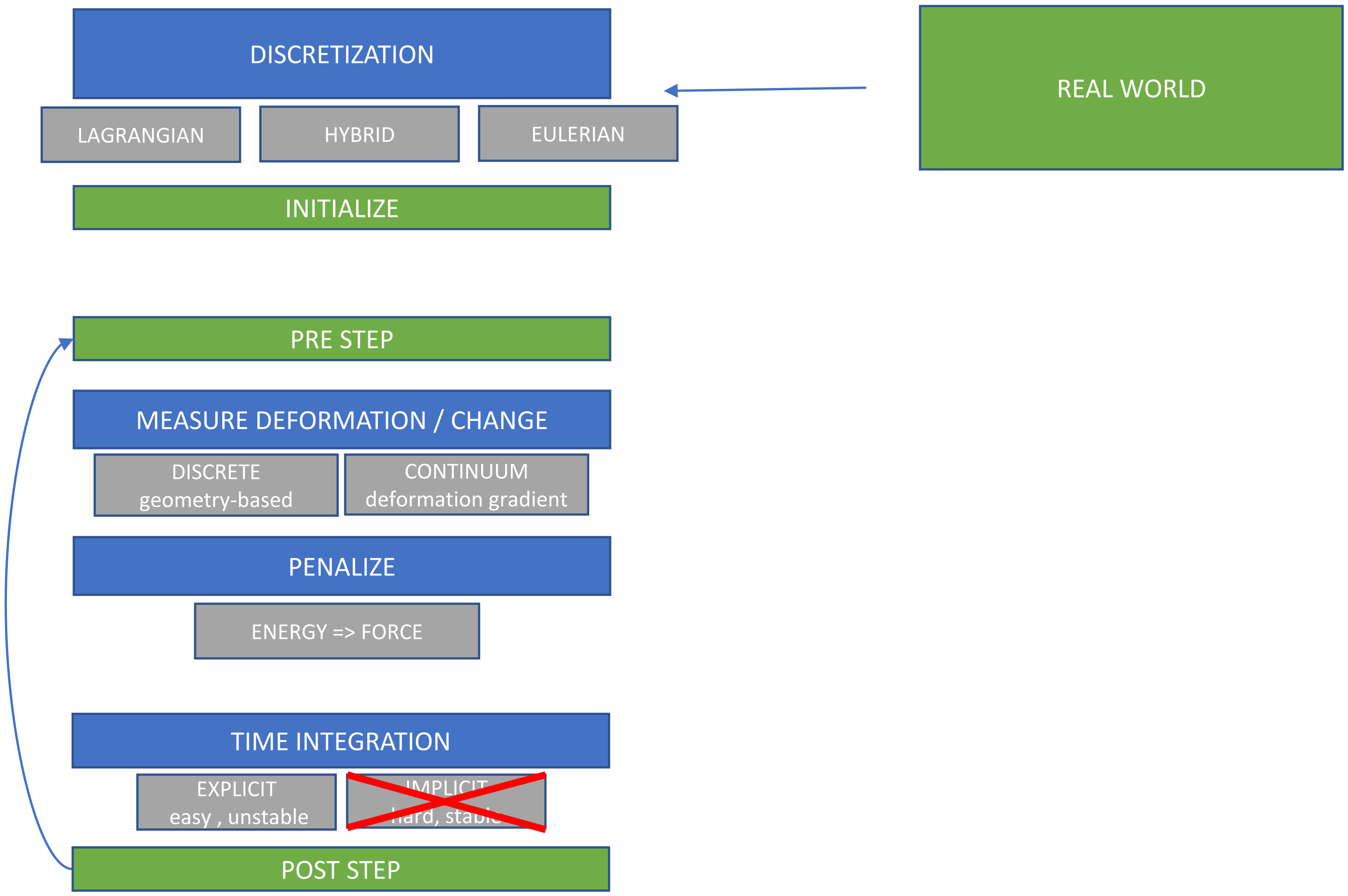


(X)PBD, nCloth, Bullet …

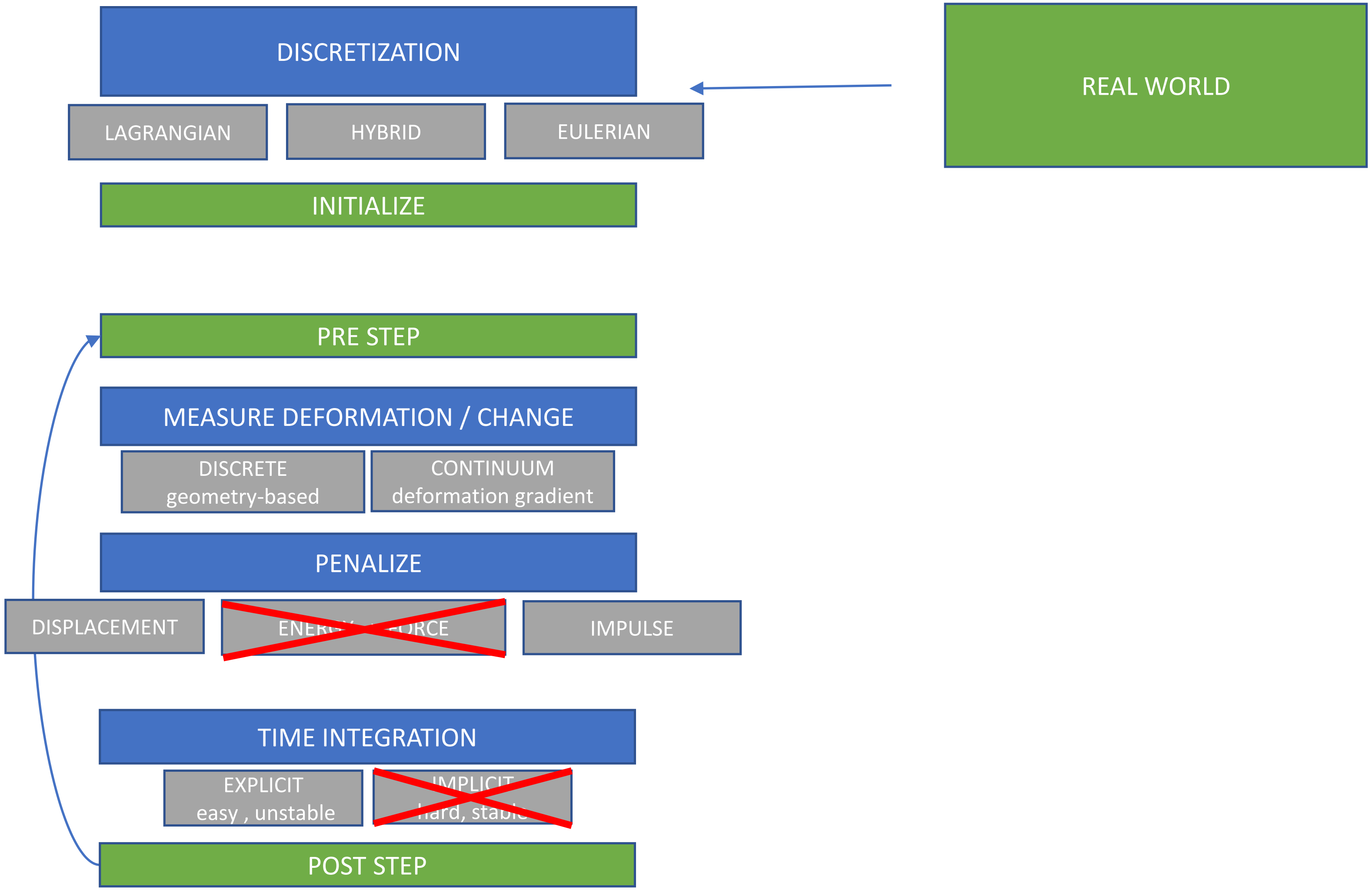
Let's simplify that stuff !



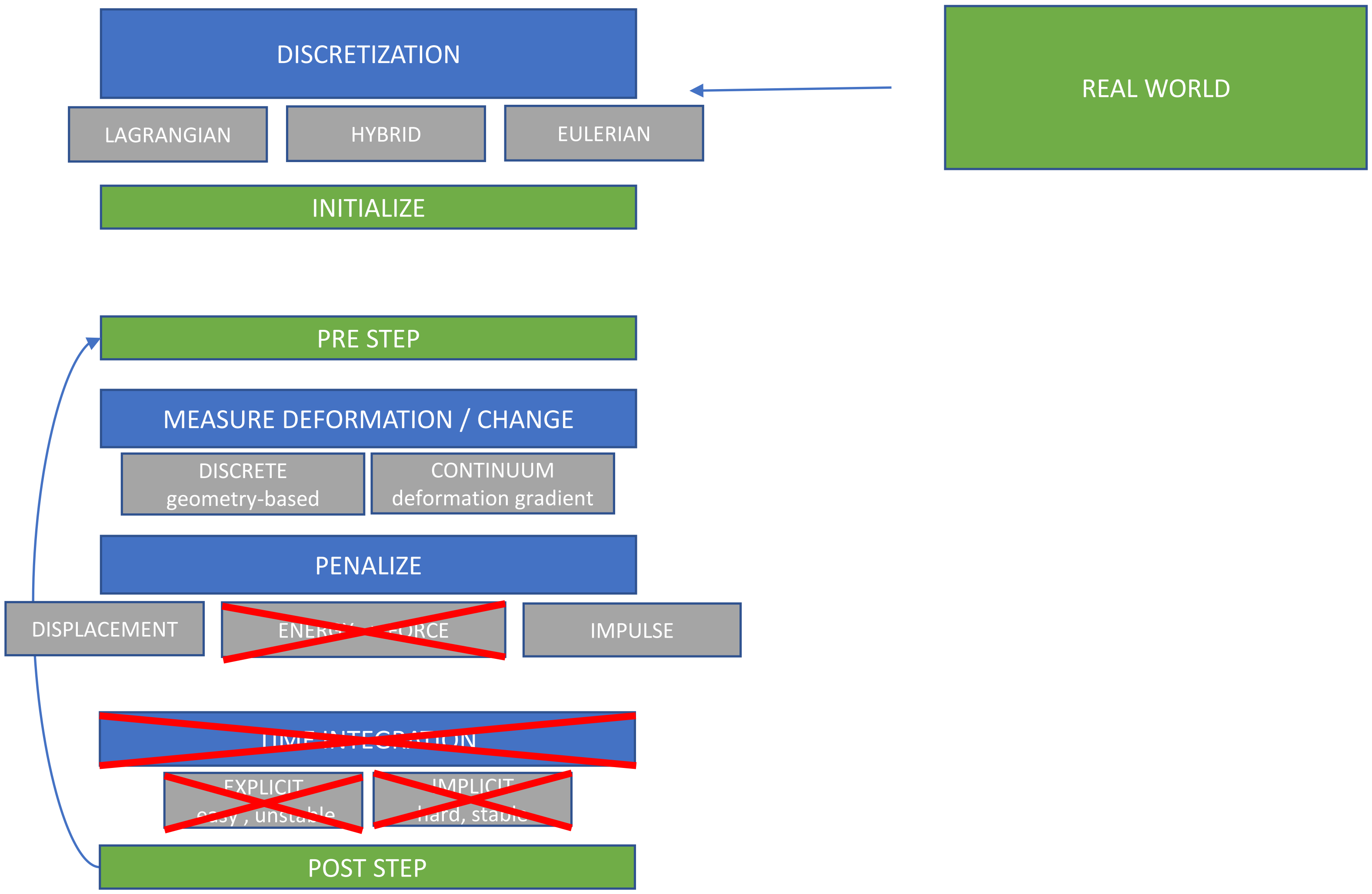
Let’s simplify that stuff !



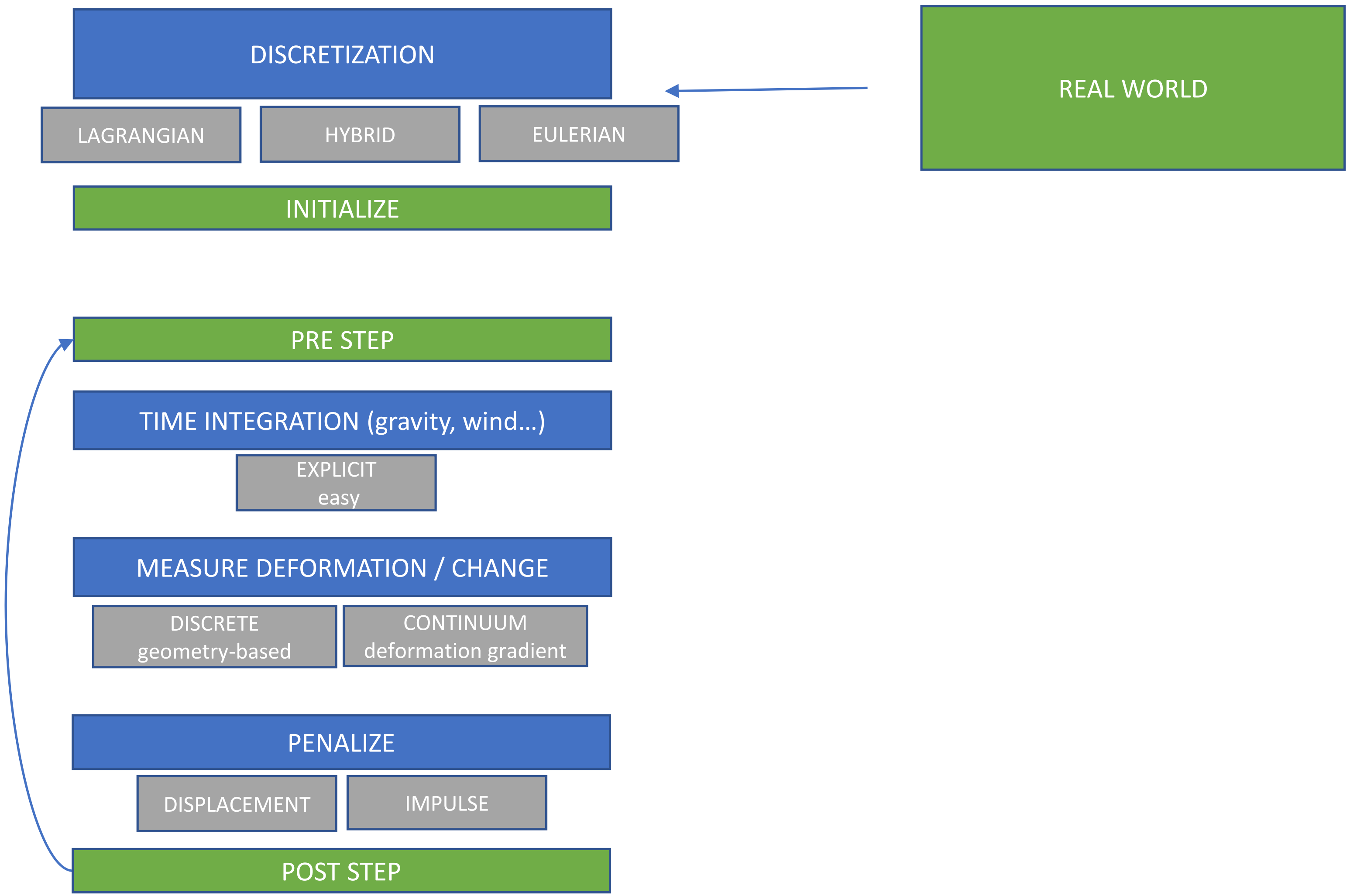
Let’s simplify that stuff !



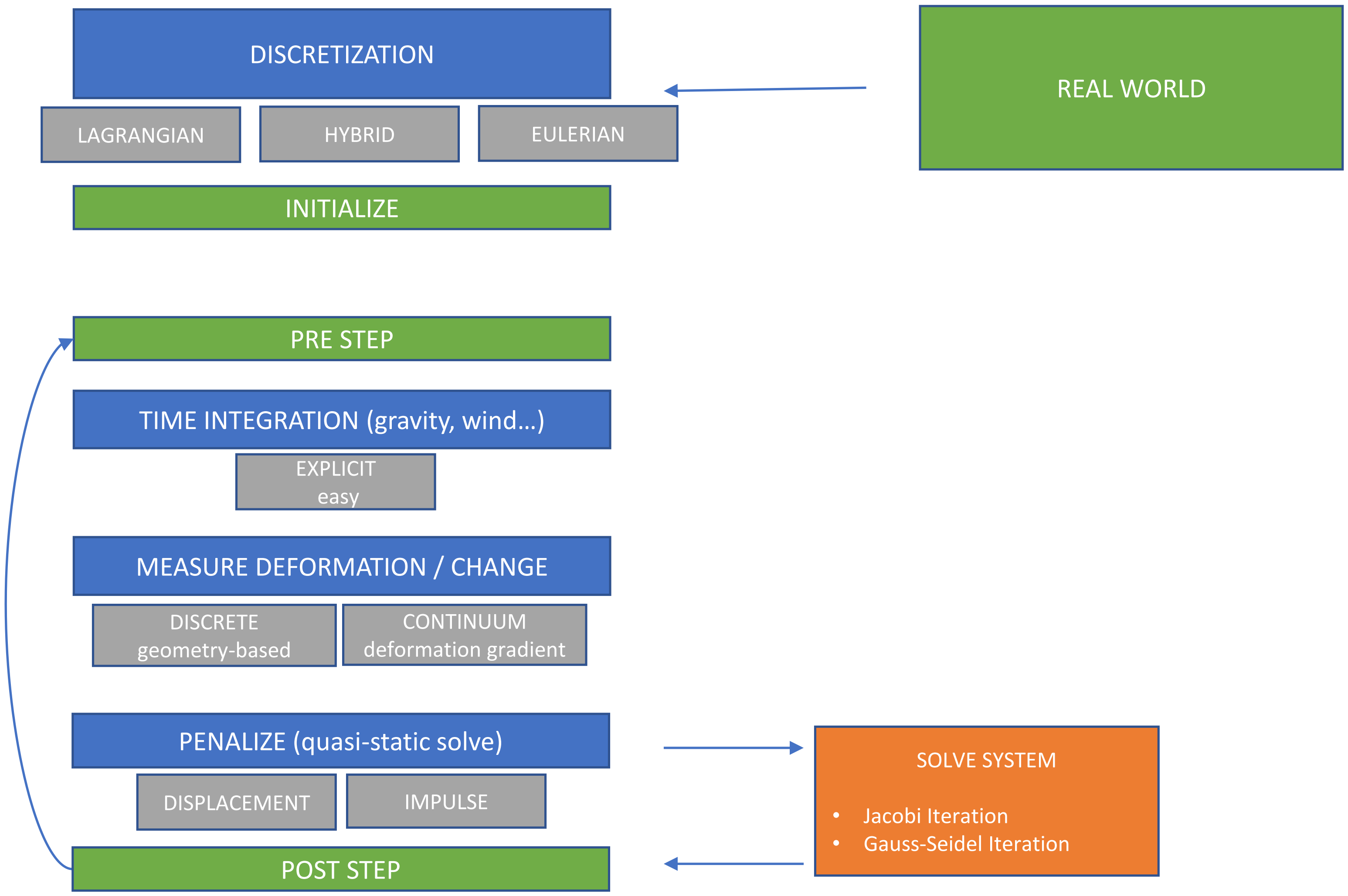
Let’s simplify that stuff !



Let’s simplify that stuff !



Let’s simplify that stuff !



WHAT DID WE LEARN !

- Three discretizations
- Discrete vs Continuous
- Time Integrators (Explicit vs Implicit)
- FEM is not about tetrahedron but continuity
- Implicit integration is hard => Workaround (PBD …)

BEYOND THIS PRESENTATION

- Penalty Method vs Lagrange Multiplier
- Direct vs Iterative Linear Solver
- Collision Detection (Discrete vs Continuous)
- Quasi-Static vs Dynamic
- Linear System Assembly
- Other Techniques (SPH / FDM)

END

RESOURCES

- Houdini PBD Training
- SIGGRAPH 2019 – PBA Course
- Ladislav Kavan’s Youtube Channel
- Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blow up." *ACM Transactions onGraphics (SIGGRAPH 1997)* (1997).

MORE DETAILS

EXPLICIT VS IMPLICIT EULER

TIME INTEGRATION

EXPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_n)$$

$$x_{n+1} = x_n + hv_{n+1}$$

h : timestep

M^{-1} : mass inverse

$f(\quad)$: force function

v_n : current velocity

x_n: current position

IMPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

v_{n+1} : next velocity

x_{n+1}: next position

EXPLICIT VS IMPLICIT EULER

TIME INTEGRATION

EXPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_n)$$

$$x_{n+1} = x_n + hv_{n+1}$$

IMPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

h : timestep

M^{-1} : mass inverse

$f(\ )$: force function

v_n : current velocity

x_n: current position

v_{n+1} : next velocity

x_{n+1}: next position

EXPLICIT VS IMPLICIT EULER

TIME INTEGRATION

EXPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_n)$$

$$x_{n+1} = x_n + hv_{n+1}$$

h : timestep

M^{-1} : mass inverse

$f(\quad)$: force function

v_n : current velocity

x_n: current position

IMPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

v_{n+1} : next velocity

x_{n+1}: next position

EXPLICIT VS IMPLICIT EULER

TIME INTEGRATION

EXPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_n)$$

$$x_{n+1} = x_n + hv_{n+1}$$

- Single line
- Conditionally stable

IMPLICIT INTEGRATOR

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

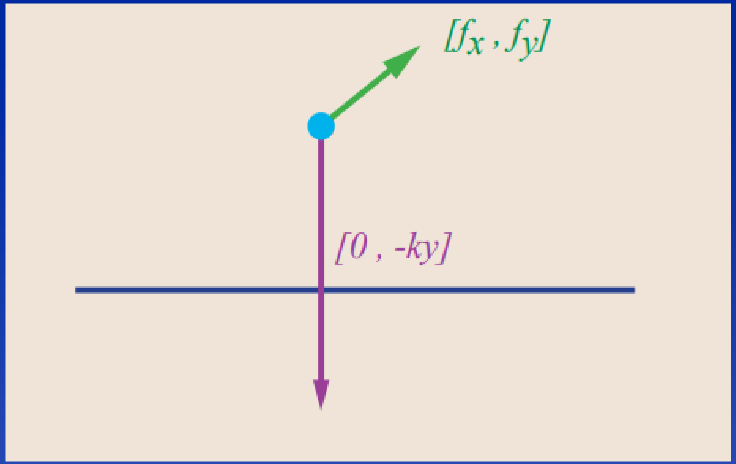
$$x_{n+1} = x_n + hv_{n+1}$$

- Solve sparse system
- Unconditionally stable

STABILITY PROOF

Example: particle-on-line

- A particle P in the plane.
- Interactive "dragging" force $[f_x, f_y]$.
- A penalty force $[0,-ky]$ tries to keep P on the x-axis.



- Suppose you want P to stay within a miniscule ε of the x-axis when you try to pull it off with a huge force f_{max}.
- How big does k have to be? How *small* must h be?

Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blowup." ACM Transactions onGraphics (SIGGRAPH 1997) (1997).

STABILITY PROOF

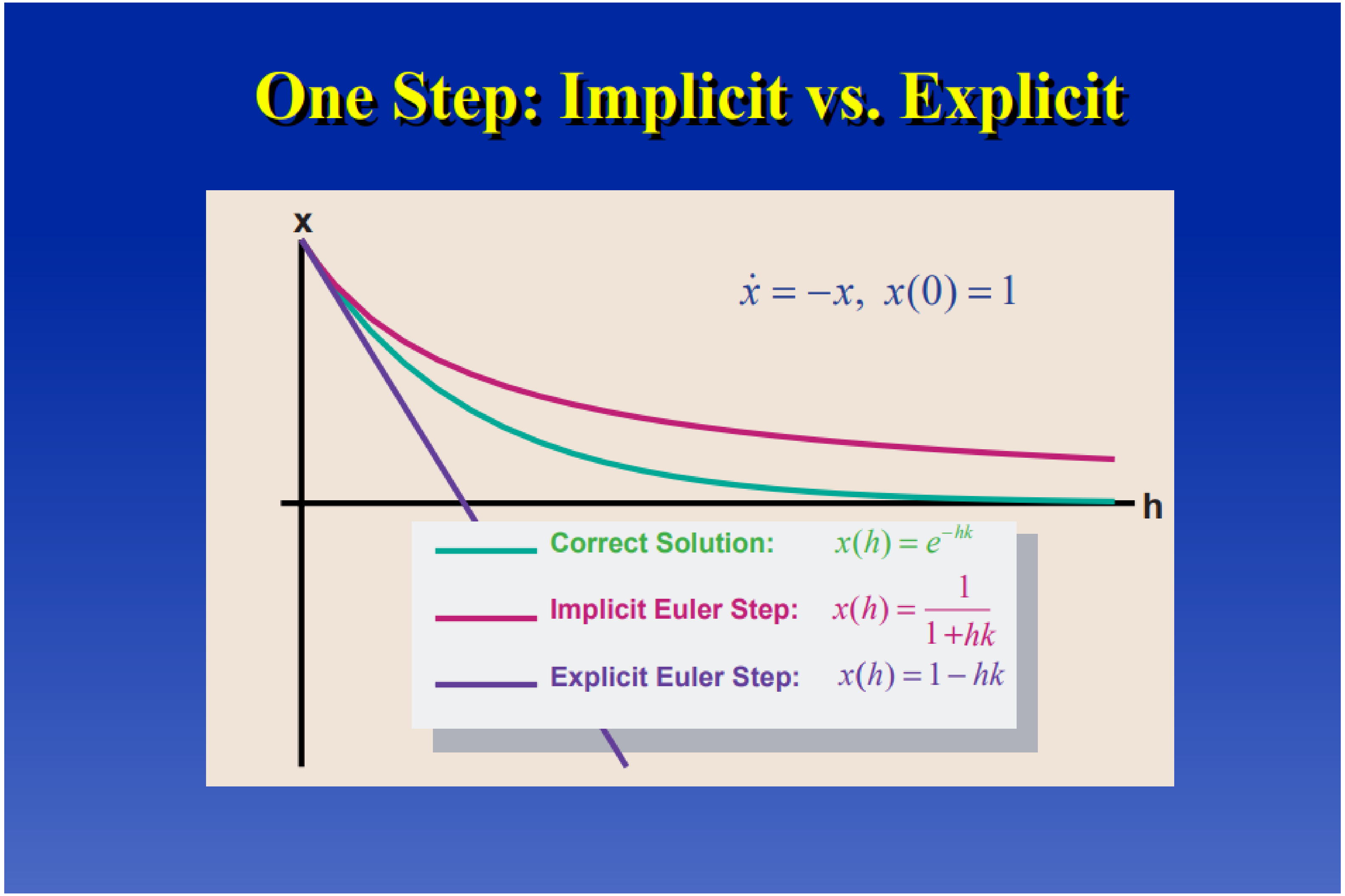
Implicit Euler for $\dot{x} = -kx$

$$
\begin{aligned}
x(t+h) &= x(t) + h\, f(x(t+h)) \\
&= x(t) - h\,k\,x(t+h) \\
&= \frac{x(t)}{1+hk}
\end{aligned}
$$

- Nonlinear: Approximate as linear, using $\partial f / \partial x$.
- Multidimensional: (sparse) matrix equation.

Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blowup." ACM Transactions onGraphics (SIGGRAPH 1997) (1997).

STABILITY PROOF



Baraff, David, and Andrew Witkin. "Implicit Methods: how to not blowup." ACM Transactions onGraphics (SIGGRAPH 1997) (1997).

SOME MORE

VARIATIONAL IMPLICIT EULER

$$\mathrm{next\_v} = \mathrm{v} + h\frac{f(next\_p)}{m}$$

$$\mathrm{next\_p} = \mathrm{p} + h\ \mathrm{next\_v}$$

MULTIPLE PARTICLES

$$v_{n+1} = v_n + hM^{-1}f(x_{n+1})$$

$$x_{n+1} = x_n + hv_{n+1}$$

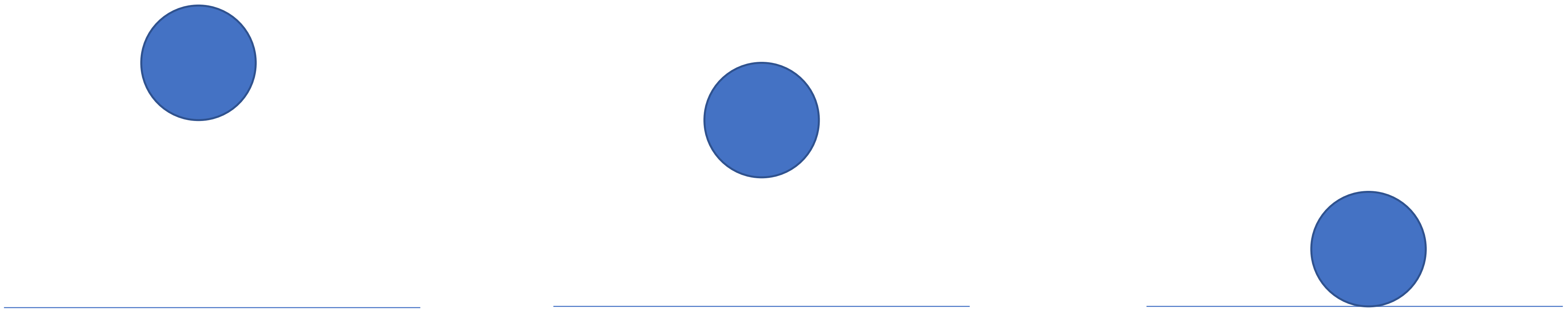
TRANSFORM INTO OPTIMIZATION PROBLEM

e(x_{n+1})=$\frac{1}{2h^2}(x_{n+1}-x_n - hv_n)^T M(x_{n+1}-x_n - hv_n) + W(x_{n+1})$

Find x_{n+1} to get a minimization of e by using the newton iteration

QUASI-STATIC VS DYNAMIC

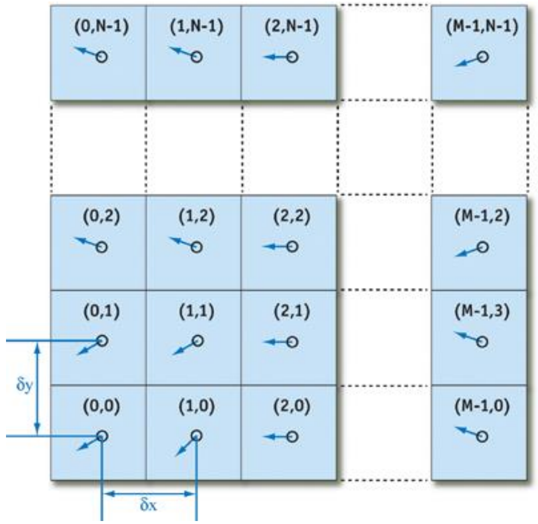
EXAMPLE : ENERGY MINIMIZATION OF GRAVITIONAL ENERGY



POTENTIAL ENERGY

OTHER MODELS

FINITE ELEMENT METHOD (FEM)

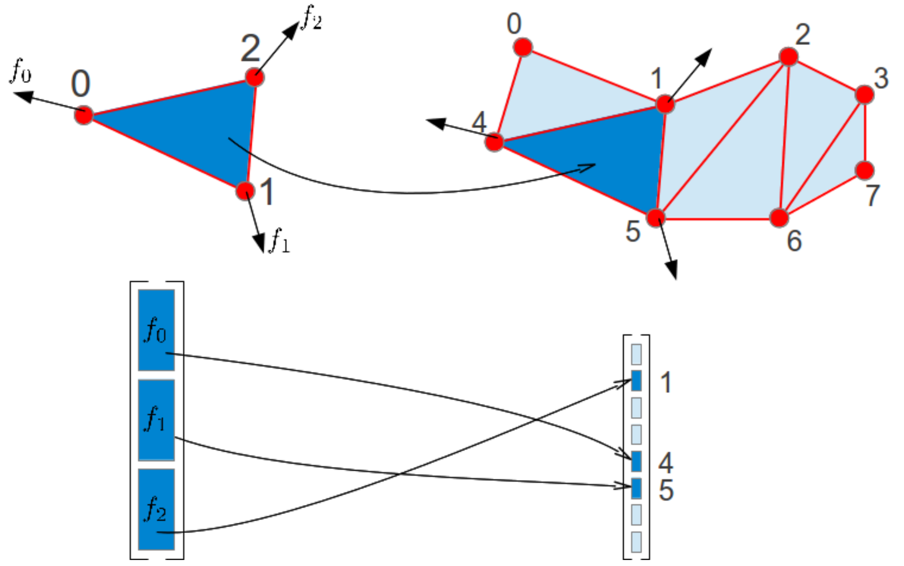


Operator	Definition	Finite Difference Form
Gradient	$\nabla p = \left(\frac{\partial p}{\partial x}, \frac{\partial p}{\partial y}\right)$	$\frac{p_{i+1,j} - p_{i-1,j}}{2\delta x}, \frac{p_{i,j+1} - p_{i,j-1}}{2\delta y}$
Divergence	$\nabla \cdot \mathbf{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$	$\frac{u_{i+1,j} - u_{i-1,j}}{2\delta x} + \frac{v_{i,j+1} - v_{i,j-1}}{2\delta y}$
Laplacian	$\nabla^2 p = \frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2}$	$\frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{(\delta x)^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{(\delta y)^2}$

Smoothed-particle hydrodynamics (SPH)

PENALTY SYSTEM ASSEMBLY

VECTOR ASSEMBLY



MATRIX ASSEMBLY