GDC March 20-24, 2023 San Francisco, CA

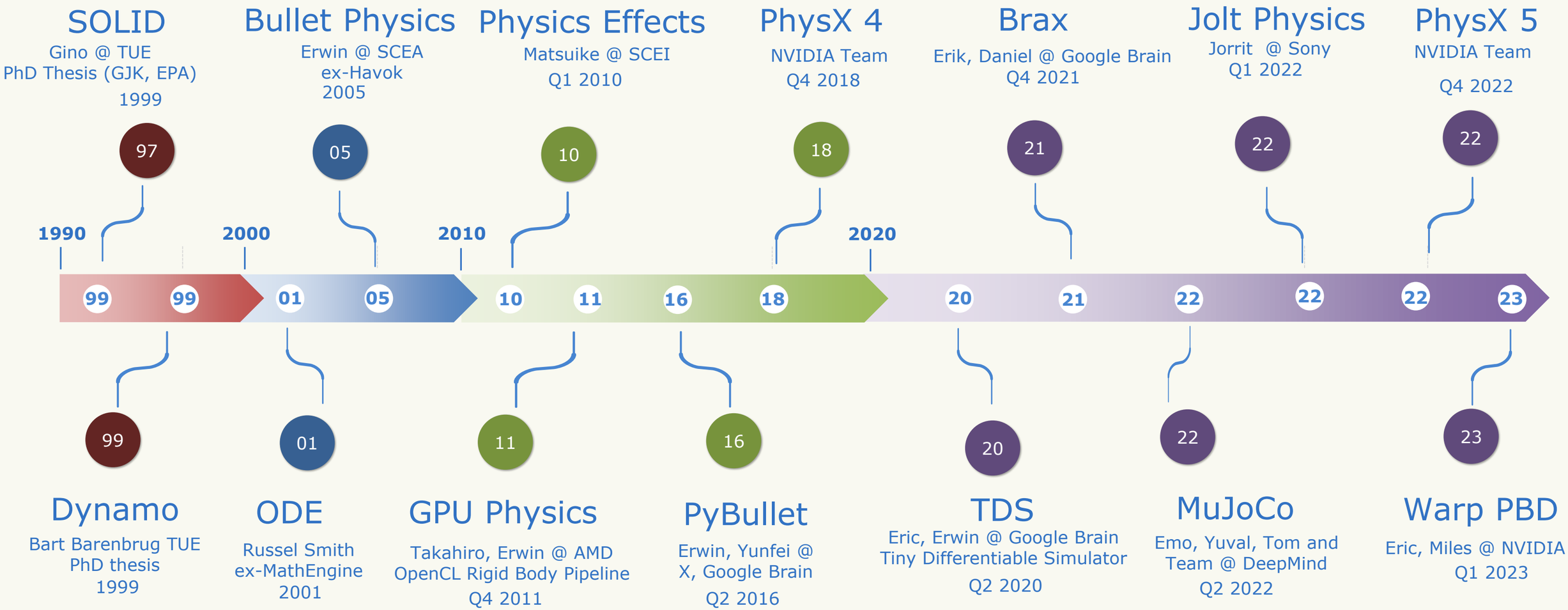
Evolution of Open Source 3D Physics Simulators

Erwin Coumans
Distinguished Engineer @ NVIDIA

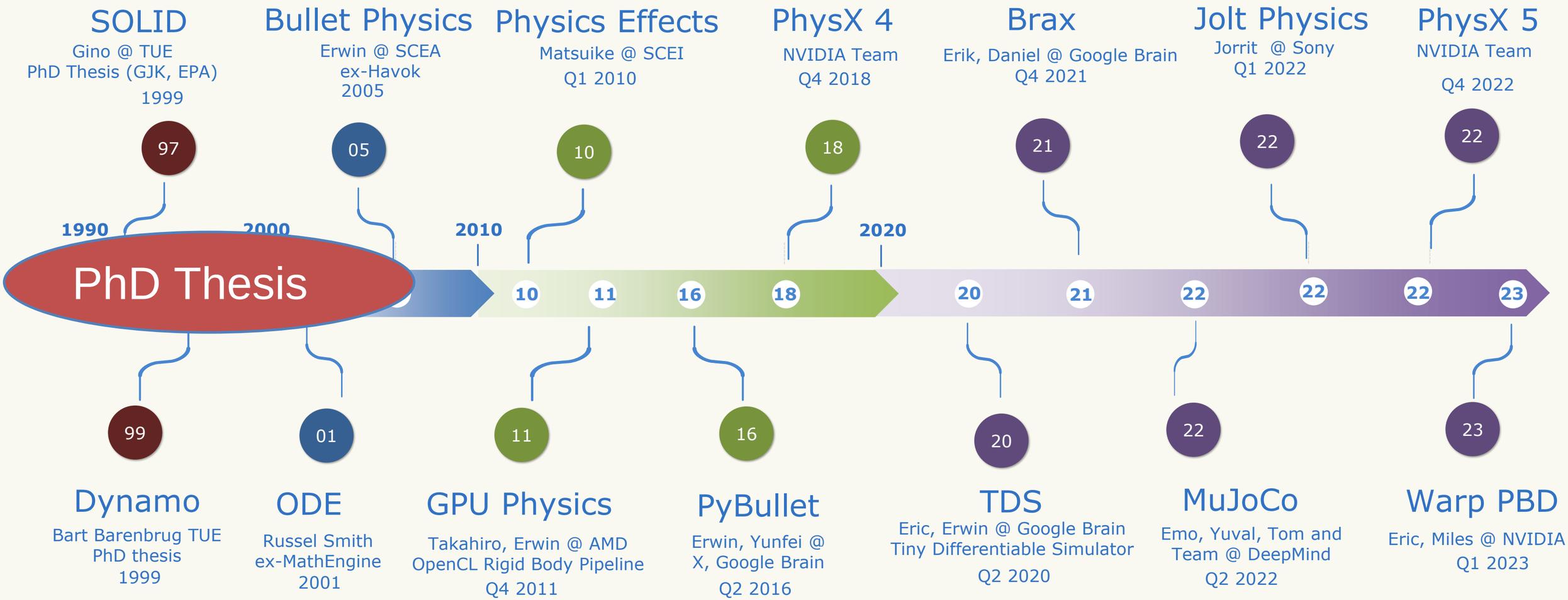
link to pptx version with vides

#GDC23

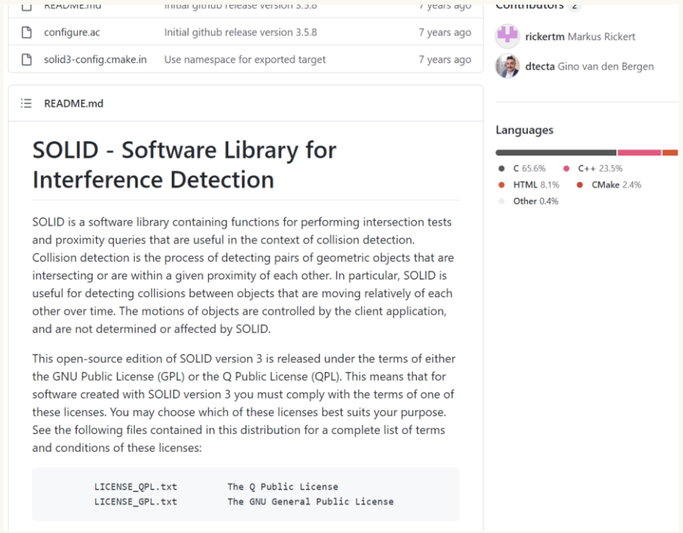
Open Source Timeline



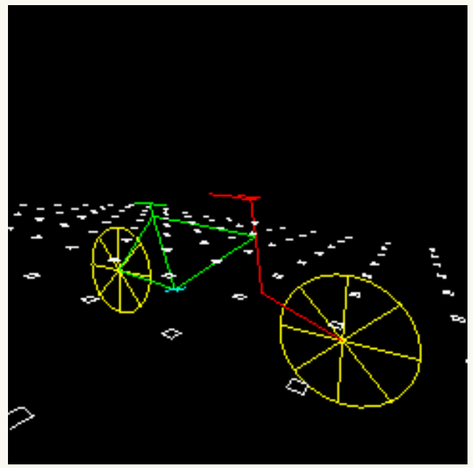
Motivation

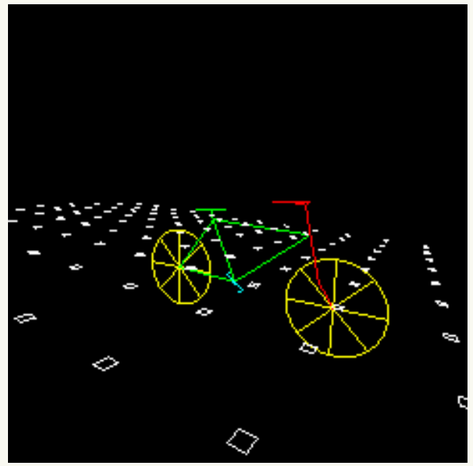

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
97
05
10
18
21
22
22
1990
2000
2010
2020
PhD Thesis
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

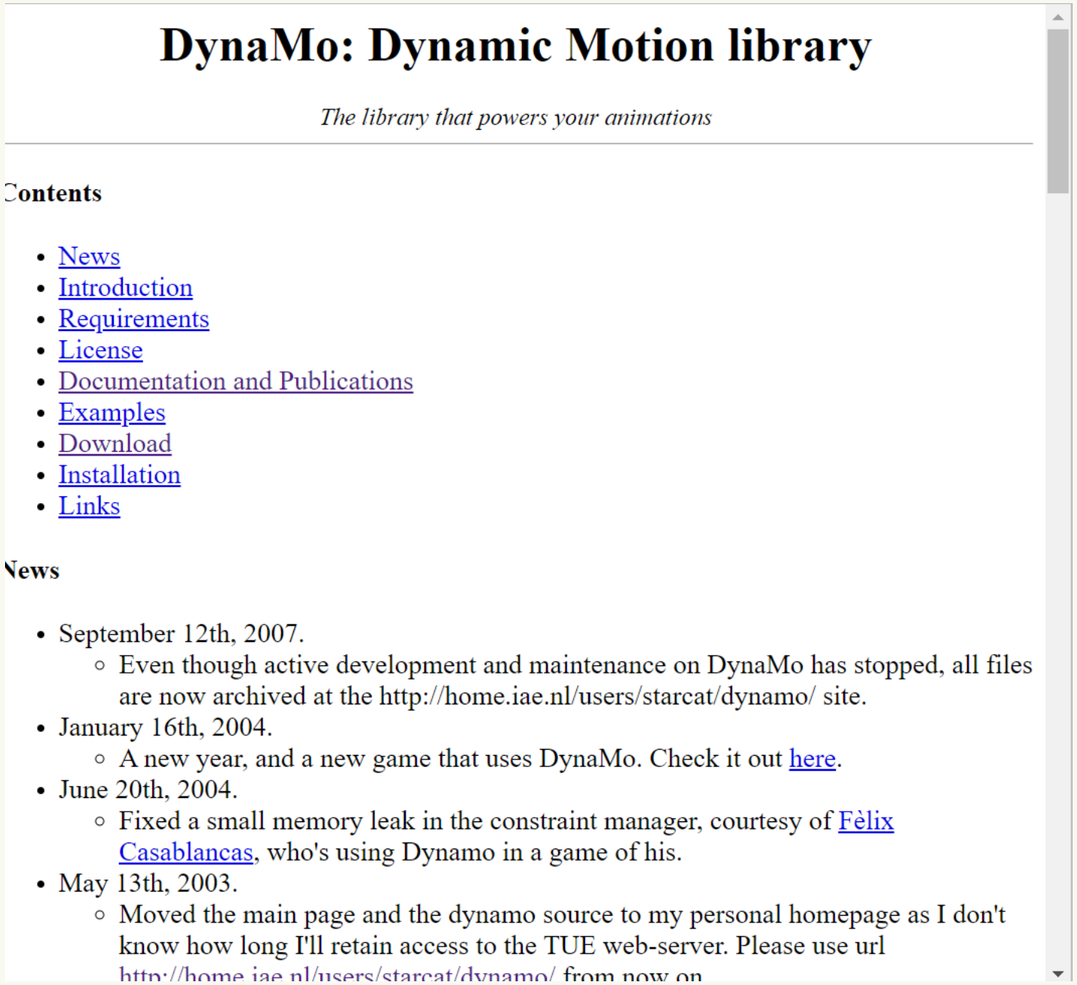
SOLID, Dynamo (1999)



https://github.com/dtecta/solid3







DynaMo: Dynamic Motion library

The library that powers your animations

Contents

- News
- Introduction
- Requirements
- License
- Documentation and Publications
- Examples
- Download
- Installation
- Links

News

- September 12th, 2007.
  - Even though active development and maintenance on DynaMo has stopped, all files are now archived at the http://home.iae.nl/users/starcat/dynamo/ site.
- January 16th, 2004.
  - A new year, and a new game that uses DynaMo. Check it out here.
- June 20th, 2004.
  - Fixed a small memory leak in the constraint manager, courtesy of Fèlix Casablancas, who's using Dynamo in a game of his.
- May 13th, 2003.
  - Moved the main page and the dynamo source to my personal homepage as I don't know how long I'll retain access to the TUE web-server. Please use url http://home.iae.nl/users/starcat/dynamo/ from now on

Dynamo PhD Thesis

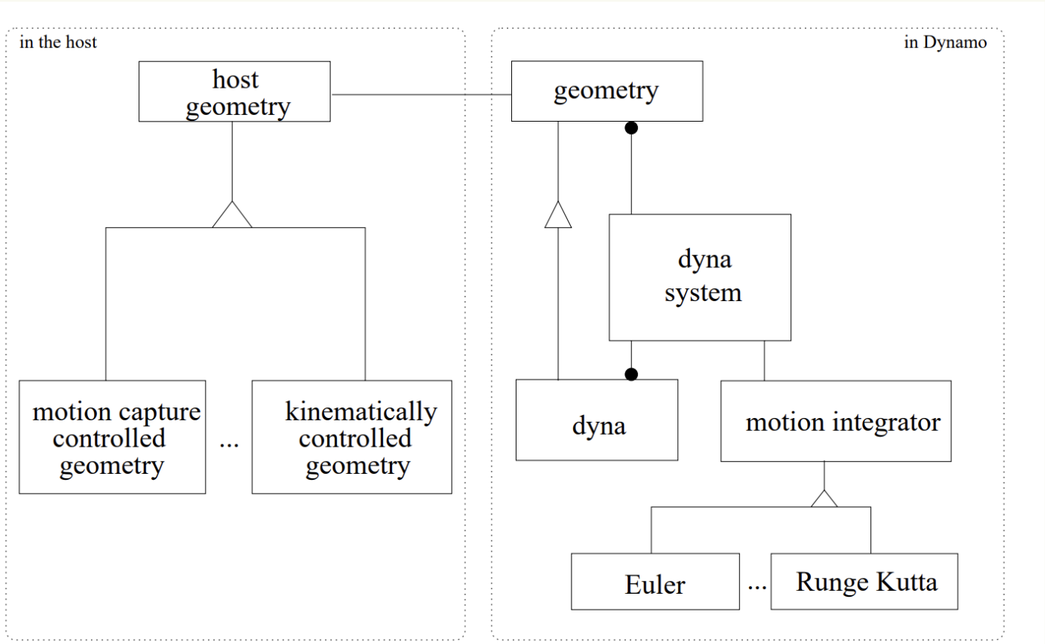
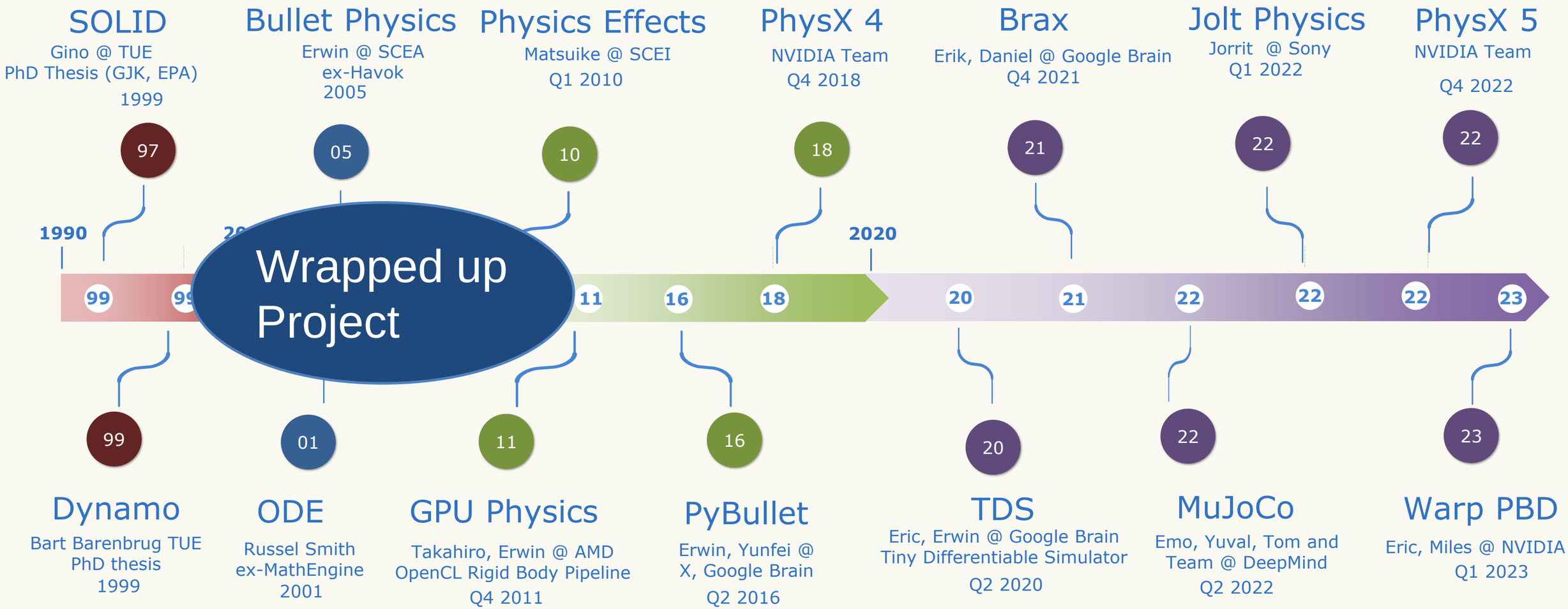


Figure 3.1: The major classes in the forward dynamics subsystem

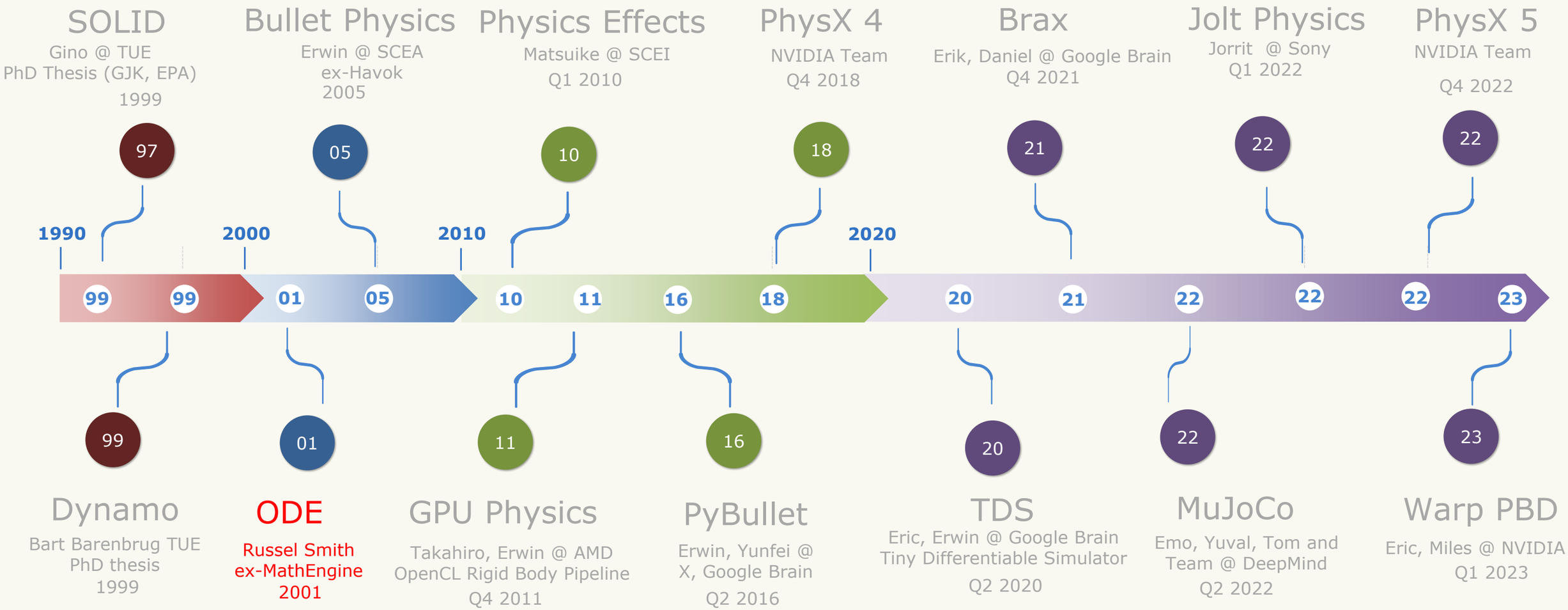
See BartThesis.pdf in the Doc folder here:

https://github.com/erwincoumans/dynamo

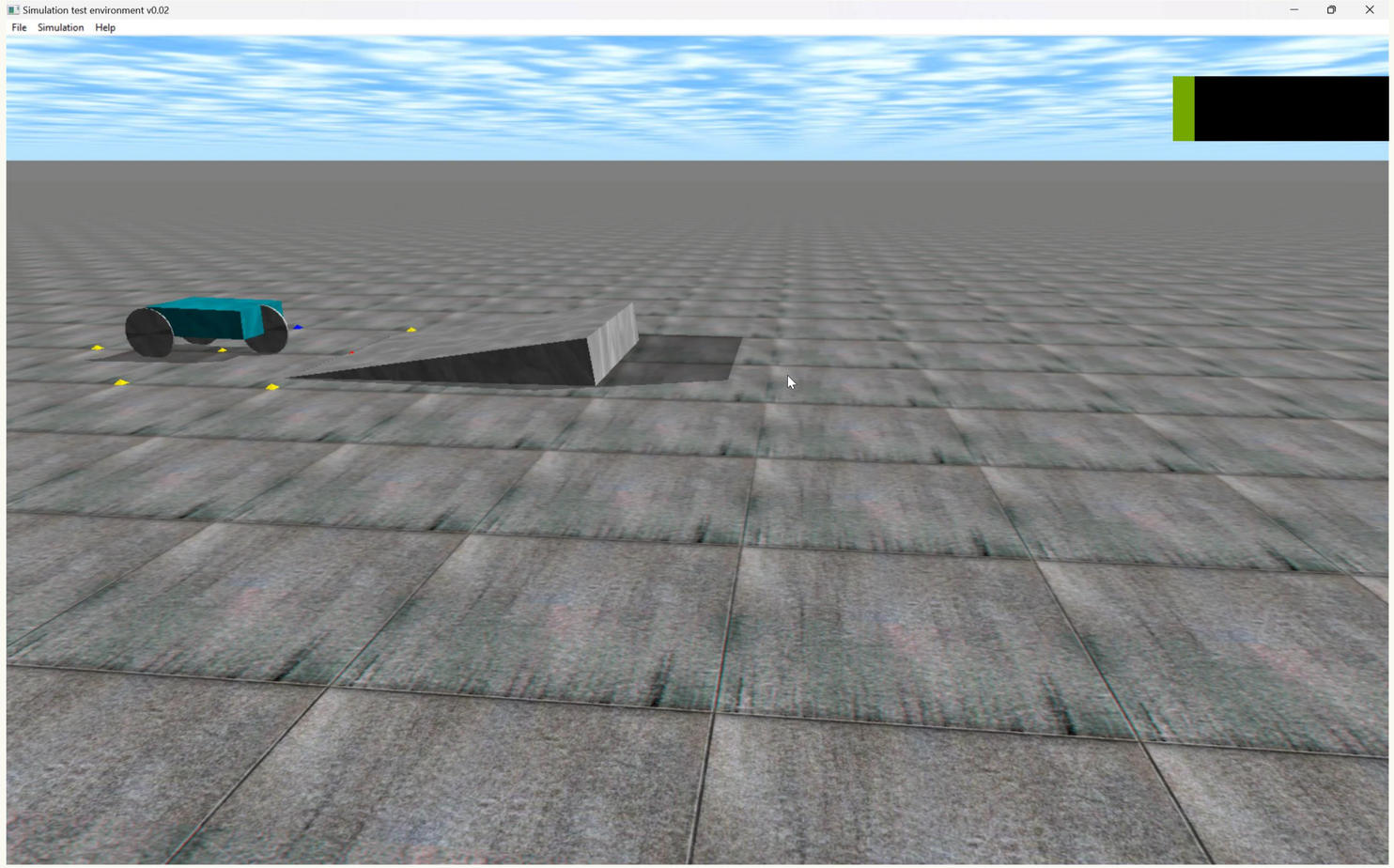
Motivation


SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2020
Wrapped up Project
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

Open Dynamics Engine



Open Dynamics Engine



Open Dynamics Engine

Russell L. Smith.
https://www.q12.org/phd/thesis/thesis.pdf
Reinforcement Learning

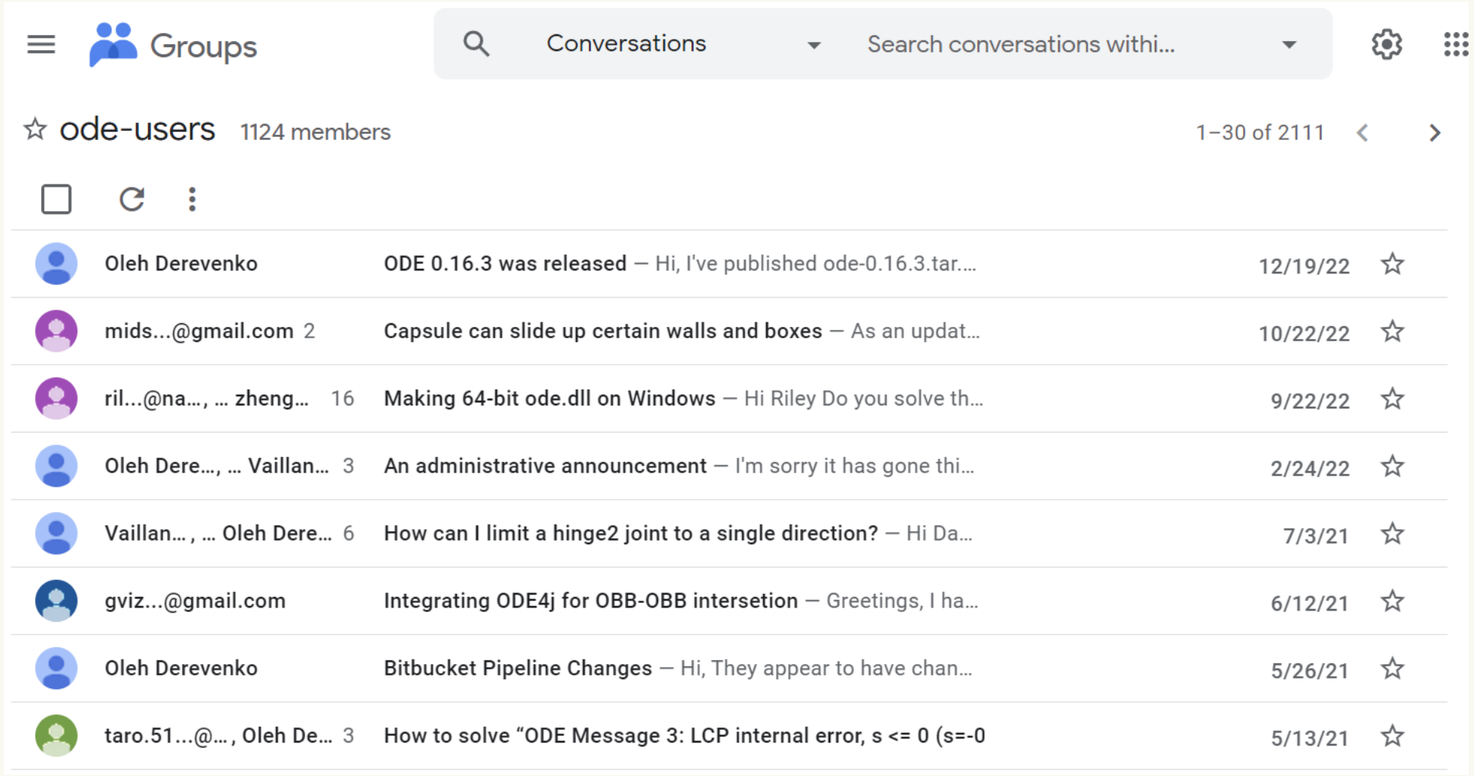
OpCode collision detection by Pierre Terdiman
ODE is maintained by Oleh Derevenko and contributors
BSD-3 License

https://www.ode.org
git clone https://bitbucket.org/odedevs/ode.git

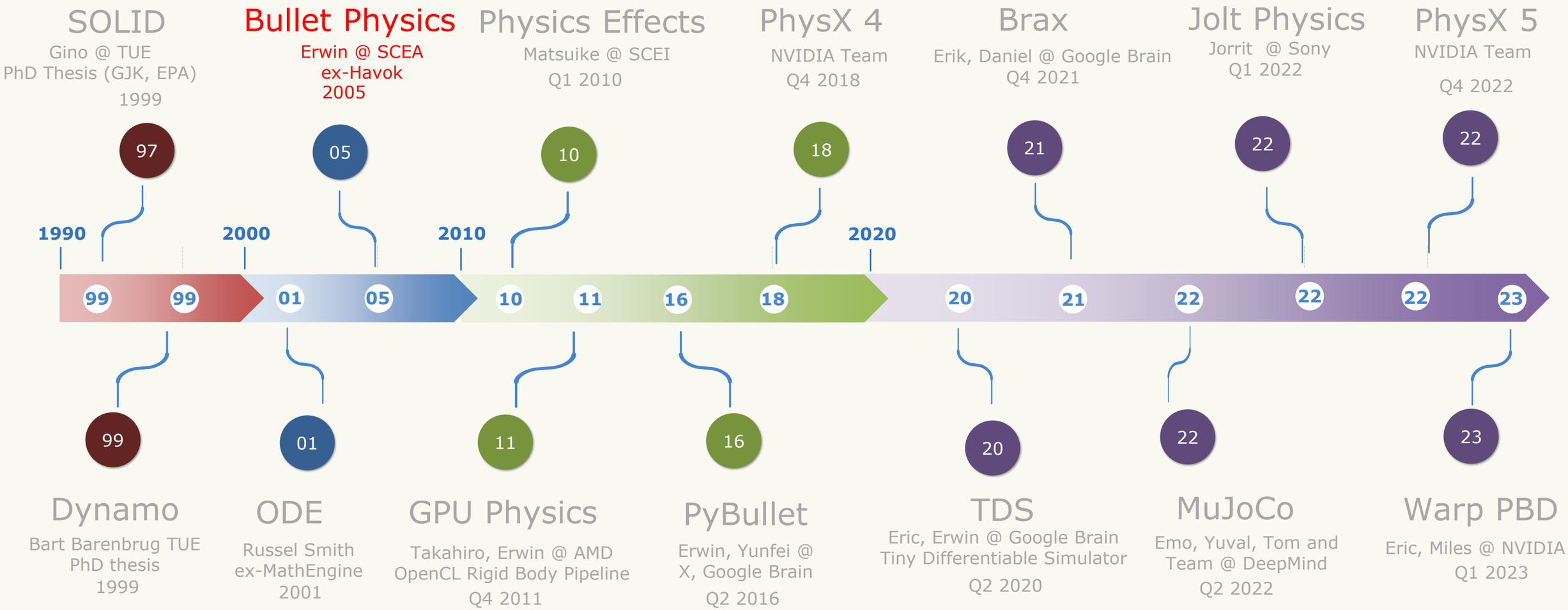
- `dSpaceCollide (space,0,&nearCallback);`
- `dWorldQuickStep (world, simstep);`
- `dBodyGetPosition(bodyId);`

Open Dynamics Engine

[ode-users - Google Groups]

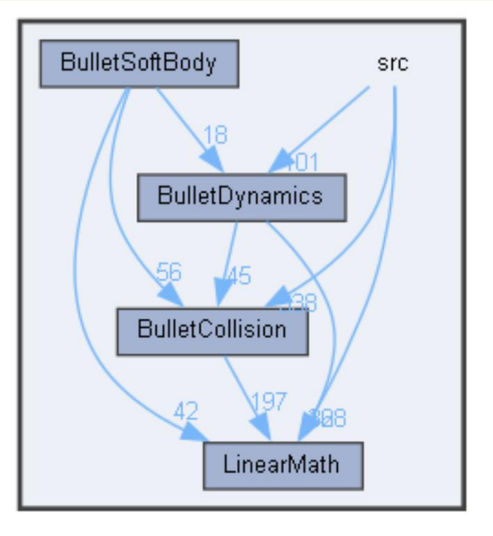


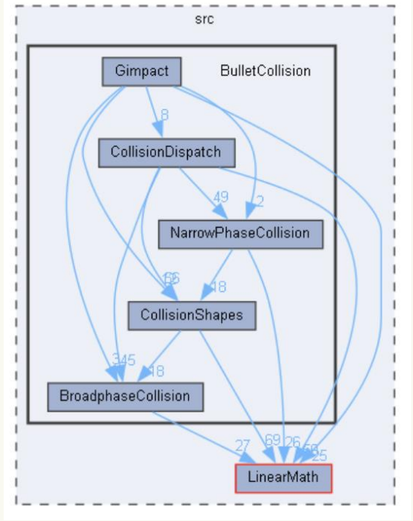
Bullet Physics

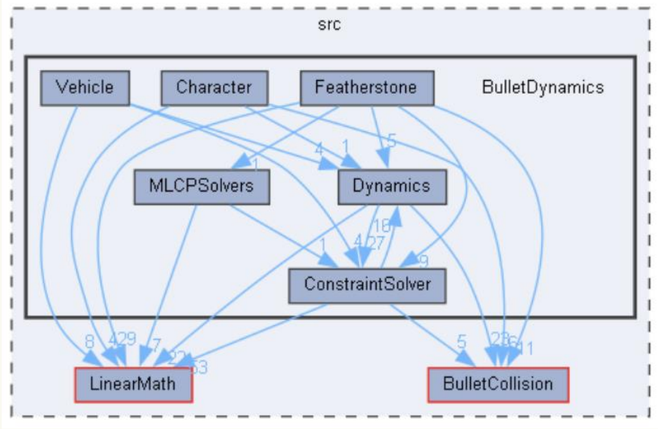


Bullet Physics

Bullet Physics







Main Page | Namespaces | Classes | Files

Bullet Documentation

Introduction

Bullet is a Collision Detection and Rigid Body Dynamics Library. The Library is Open Source and free for commercial use, under the ZLib license (http://openso

The main documentation is Bullet_User_Manual.pdf, included in the source code distribution. There is the Physics Forum for feedback and general Collision Det

Installation

Step 1: Download

You can download the **Bullet** Physics Library from the github repository: https://github.com/bulletphysics/bullet3/releases

Step 2: Building

Bullet has multiple build systems, including premake, cmake and autotools. Premake and cmake support all platforms. Premake is included in the Bullet/build fo create Microsoft Visual Studio projects. On Mac OSX and Linux you can open a terminal and generate Makefile, codeblocks or Xcode4 projects: cd Bullet/build . ./premake4_osx xcode4 cd Bullet/build/gmake make

An alternative to premake is cmake. You can download cmake from http://www.cmake.org cmake can autogenerate projectfiles for Microsoft Visual Studio, Apple interface and choose the options and generate projectfiles. You can also use cmake in the command-line. Here are some examples for various platforms: cmake recommended, you can also use autotools for UNIX: ./autogen.sh ./configure to create a Makefile and then run make.

Step 3: Testing demos

Try to run and experiment with BasicDemo executable as a starting point. **Bullet** can be used in several ways, as Full Rigid Body simulation, as Collision Detect seen in this documentation under Directories

Step 4: Integrating in your application, full Rigid Body and Soft Body simulation

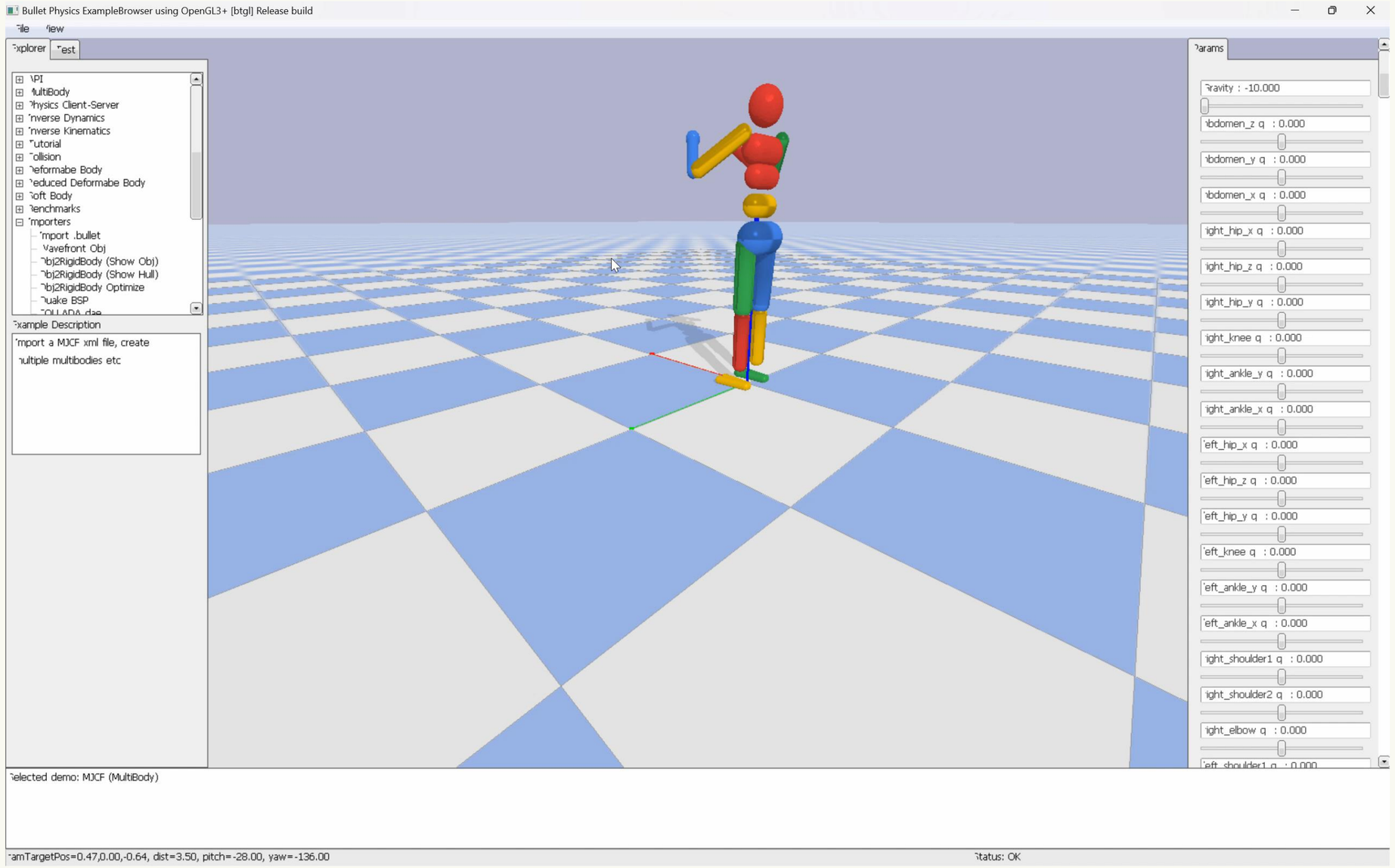
Check out BasicDemo how to create a **btDynamicsWorld**, **btRigidBody** and **btCollisionShape**, Stepping the simulation and synchronizing your graphics objec **btSoftRigidDynamicsWorld**.

Step 5 : Integrate the Collision Detection Library (without Dynamics and other Extras)

Bullet Collision Detection can also be used without the Dynamics/Extras. Check out **btCollisionWorld** and **btCollisionObject**, and the CollisionInterfaceDemo.

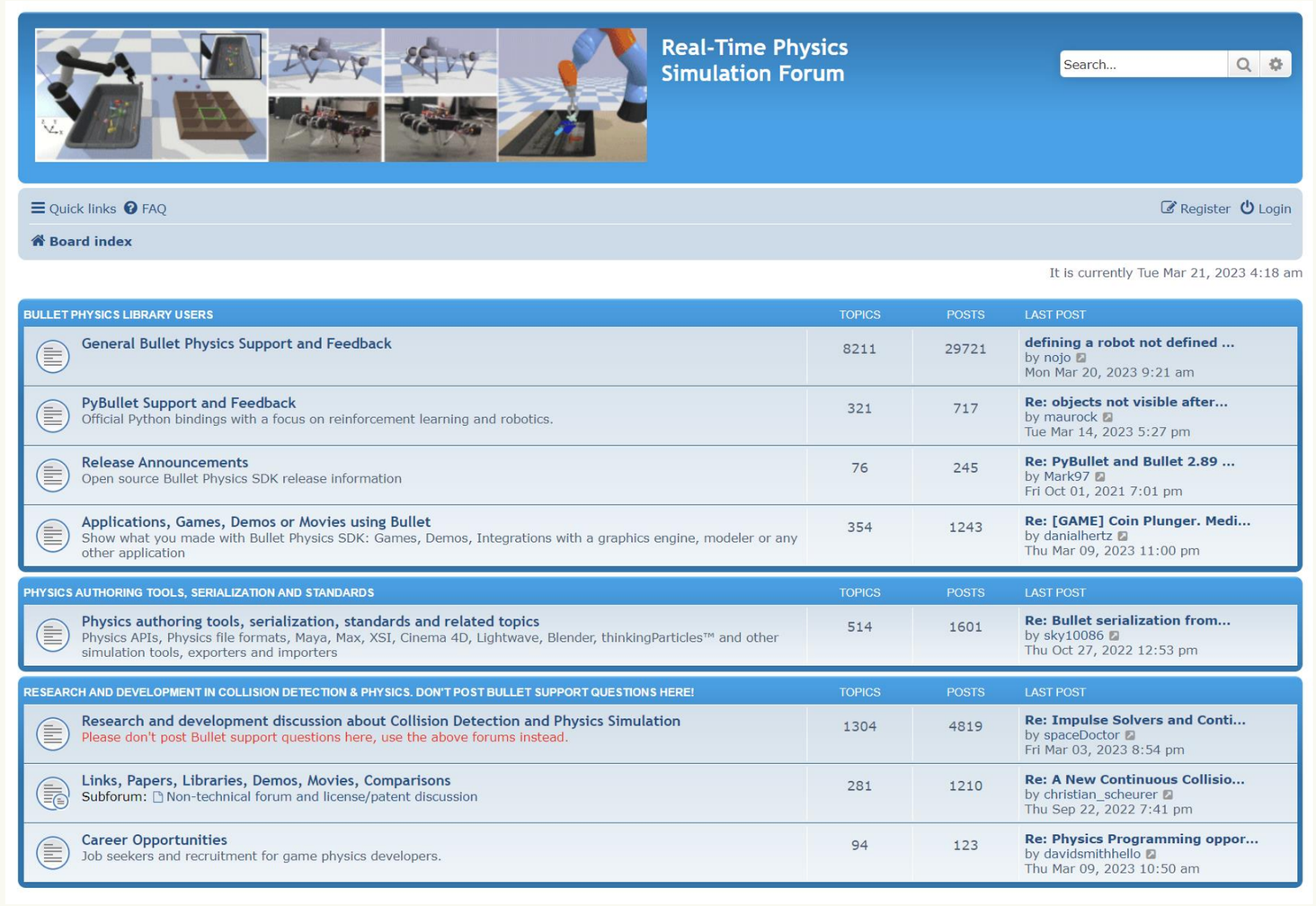
Step 6 : Use Snippets like the GJK Closest Point calculation.

Bullet has been designed in a modular way keeping dependencies to a minimum. The ConvexHullDistance demo demonstrates direct use of **btGjkPairDetecto**

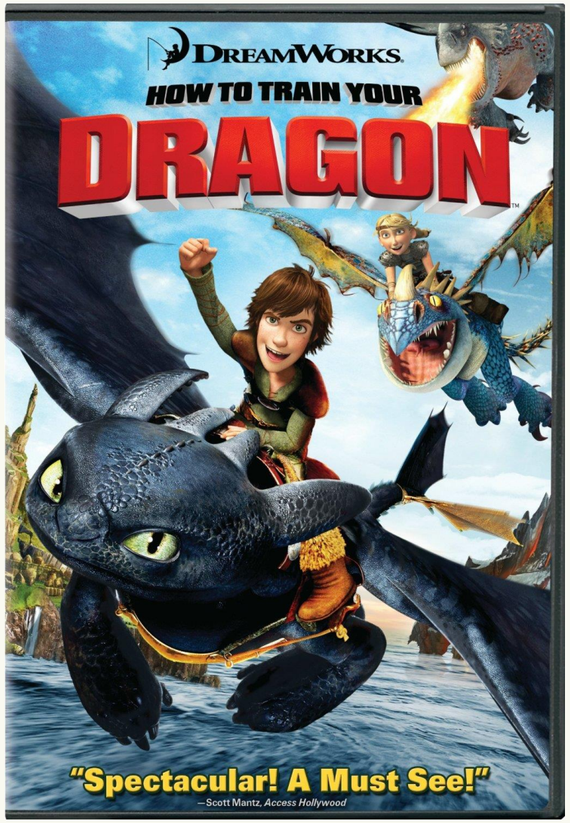


https://github.com/bulletphysics/bullet3

Physics Simulation Forum

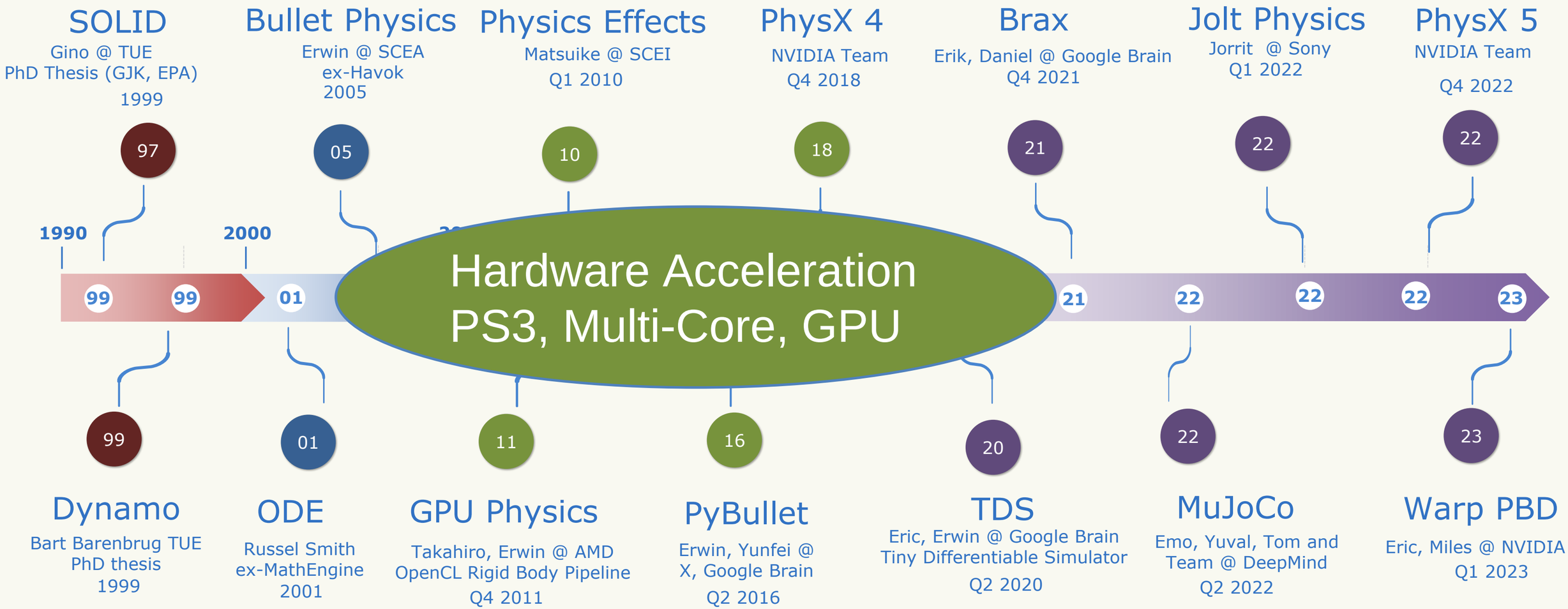


Bullet in Games and VFX

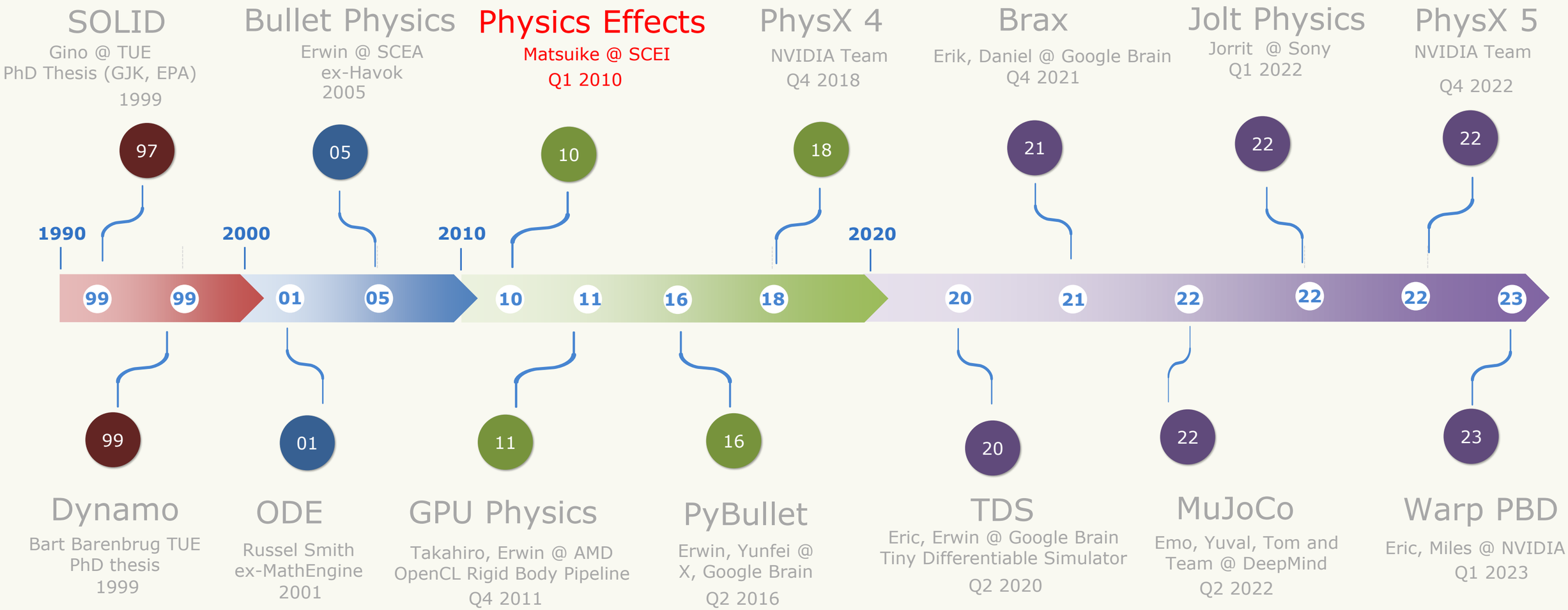


User Feedback defines the Product

Motivation


SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
97
05
10
18
21
22
22
1990
2000
99
99
01
21
22
22
22
23
Hardware Acceleration
PS3, Multi-Core, GPU
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

Physics Effects

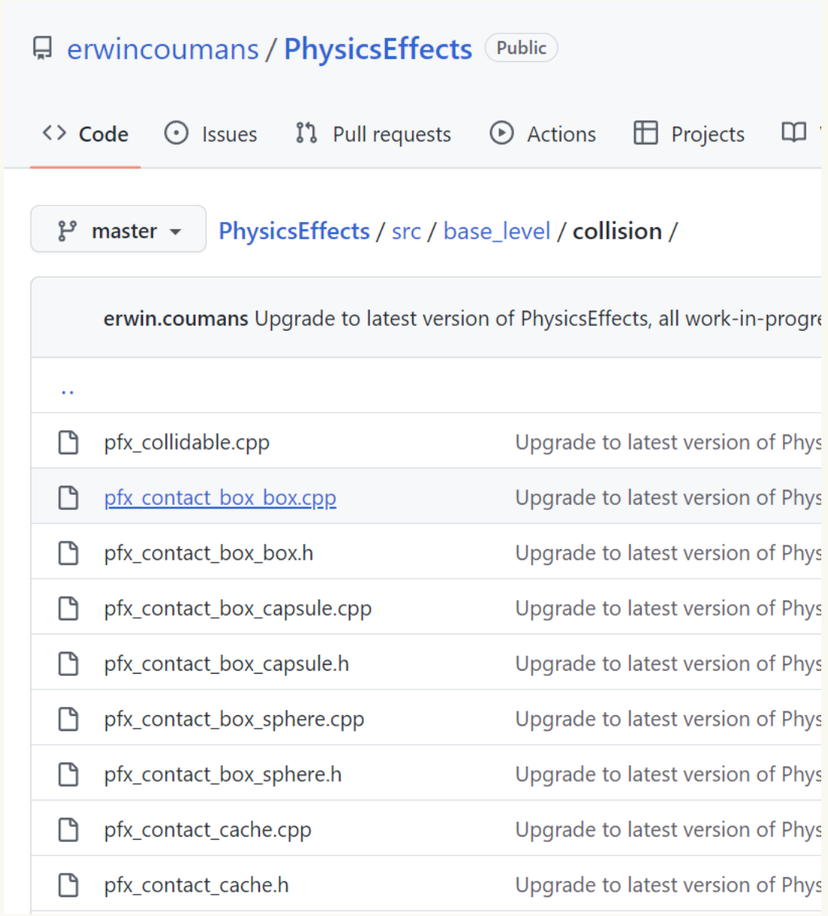


Physics Effects ‘Acceleration’

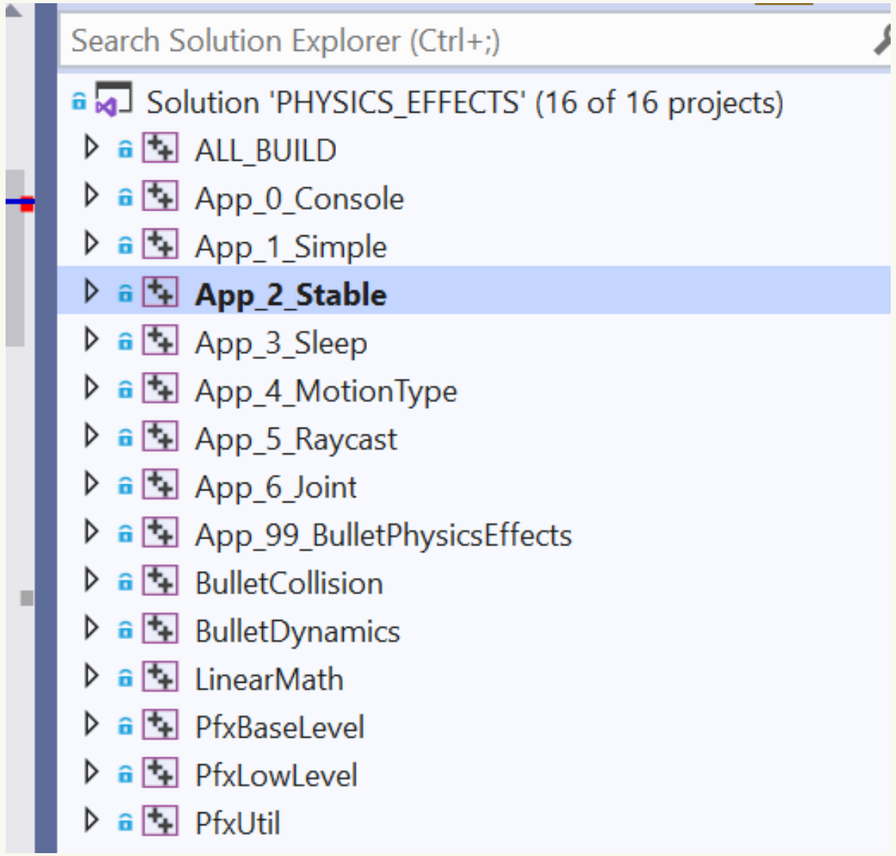
Immediate Mode Physics
base level + low level
PS3 SPU Implementation

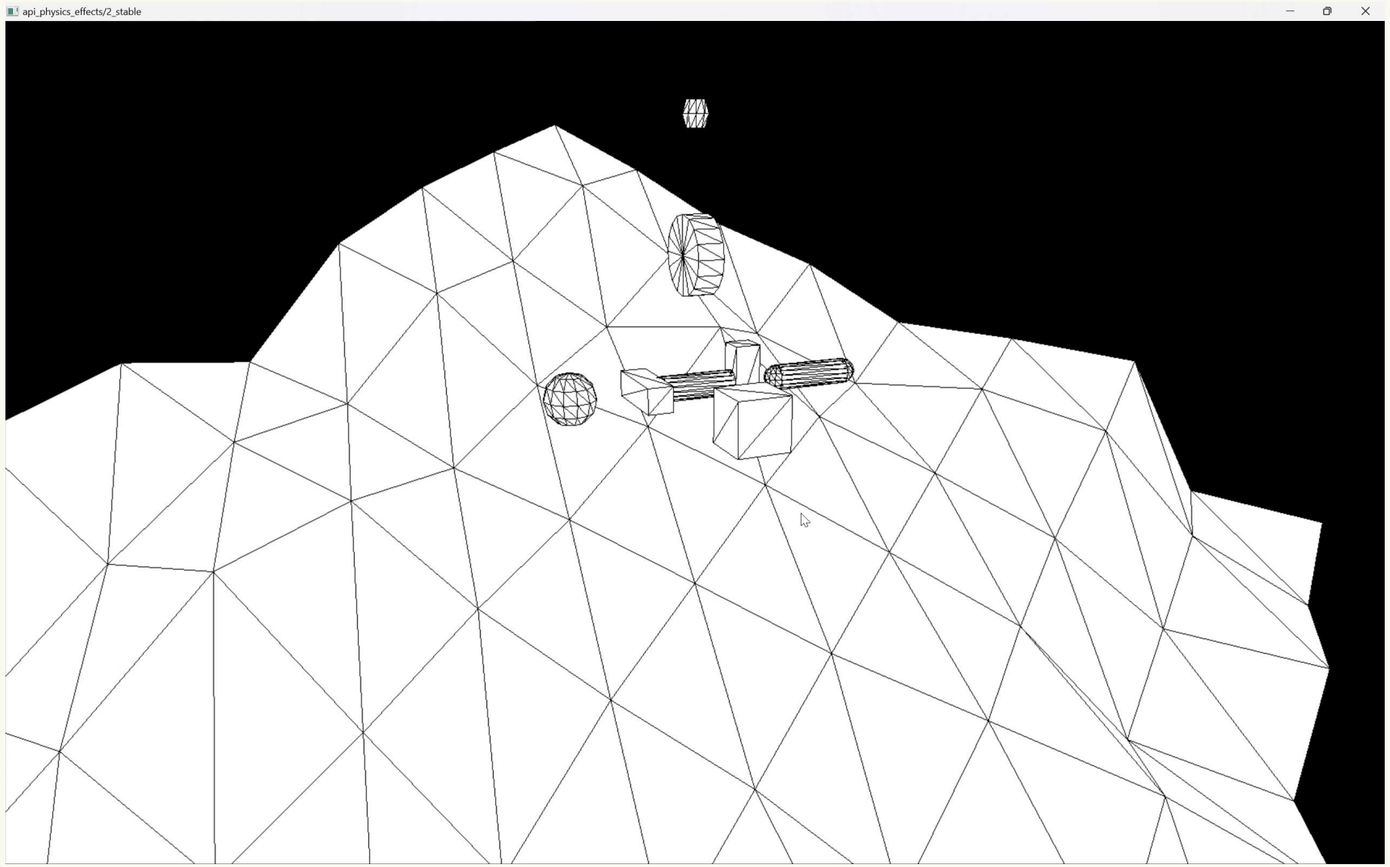
COLLADA Physics format
Unreal Engine Integration

Long term goal:
Open physics cross-platform standardization



Physics Effects





https://github.com/erwincoumans/PhysicsEffects

COLLADA Physics

Physics Engine	PhysX	Havok	ODE	Bullet
Box, capsule, sphere,cylinder	Yes	Yes	Yes	Yes
Convex mesh	Yes	Yes	Partial	Yes
Concave Triangle mesh	Yes	Yes	Yes	Yes
Compound shapes	Yes	Yes	Yes	Yes
Recursive Shapes	No	Yes	No	Yes
Rigid Body mass, inertia	Yes	Yes	Yes	Yes
Friction, Restitution	Yes	Yes	Yes	Yes
Center of mass offset	Yes	Yes	Yes	Yes
Hinge,Ball Socket, Slider	Yes	Yes	Yes	Yes
Motors, Limits	Yes	Yes	Yes	Partial
Generic 6DOF Constraint	Yes	TBD	TBD	Partial

Table 2. Physics Engines common features

3D Tool	Collision shapes	Rigidbody	Constraints
Maya 8 + COLLADA Maya [Feeling]	All primitives, compound, convex & concave mesh	Yes	Yes
3ds Max 8 + PhysX 2.6.2 plugin [Ageia]	Yes	Yes	Yes
XSI	Planned	Planned	Planned
Blender 2.43	Box, sphere, cylinder, convex and concave mesh	Yes	Partial

Table 3. COLLADA Physics tools support

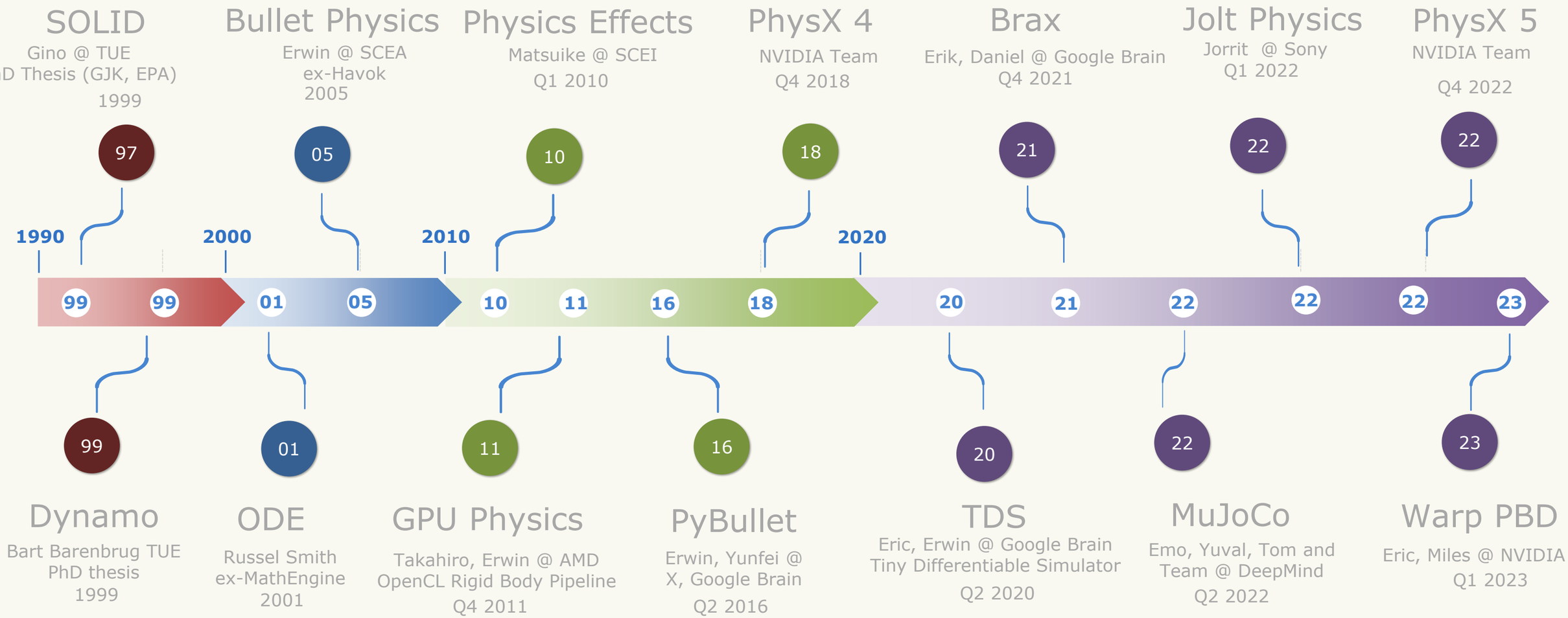
```
<rigid_body name="RigidBody1" sid="RigidBody1">
 <technique_common>
  <dynamic sid="RigidBody1">1</dynamic>
  <instance_physics_material url="#RigidBody1-material"/>
   <shape>
        <hollow>false</hollow>
        <sphere>
        <radius>10.000000</radius>
        </sphere>
   </shape>
  </technique_common>
</rigid_body>
```

```
<?xml version="1.0" encoding="utf-8"?>
<COLLADA
xmlns="http://www.collada.org/2005/11/COLLADASchema"
version="1.4.1">
 <library_physics_scenes>
  <physics_scene id="NativePhysicsScene">
   <instance_physics_model url="#PhysicsModel1">
    <instance_rigid_body target="#RigidBody1"
body="graphicsNode1">
     <technique_common/>
    </instance_rigid_body>
   </instance_physics_model>
   <technique_common>
    <gravity>0 -9.81 0</gravity>
   </technique_common>
  </physics_scene>
 </library_physics_scenes>
 <scene>
  <instance_physics_scene url="#NativePhysicsScene"/>
  <instance_visual_scene url="#VisualSceneNode"/>
 </scene>
</COLLADA>
```

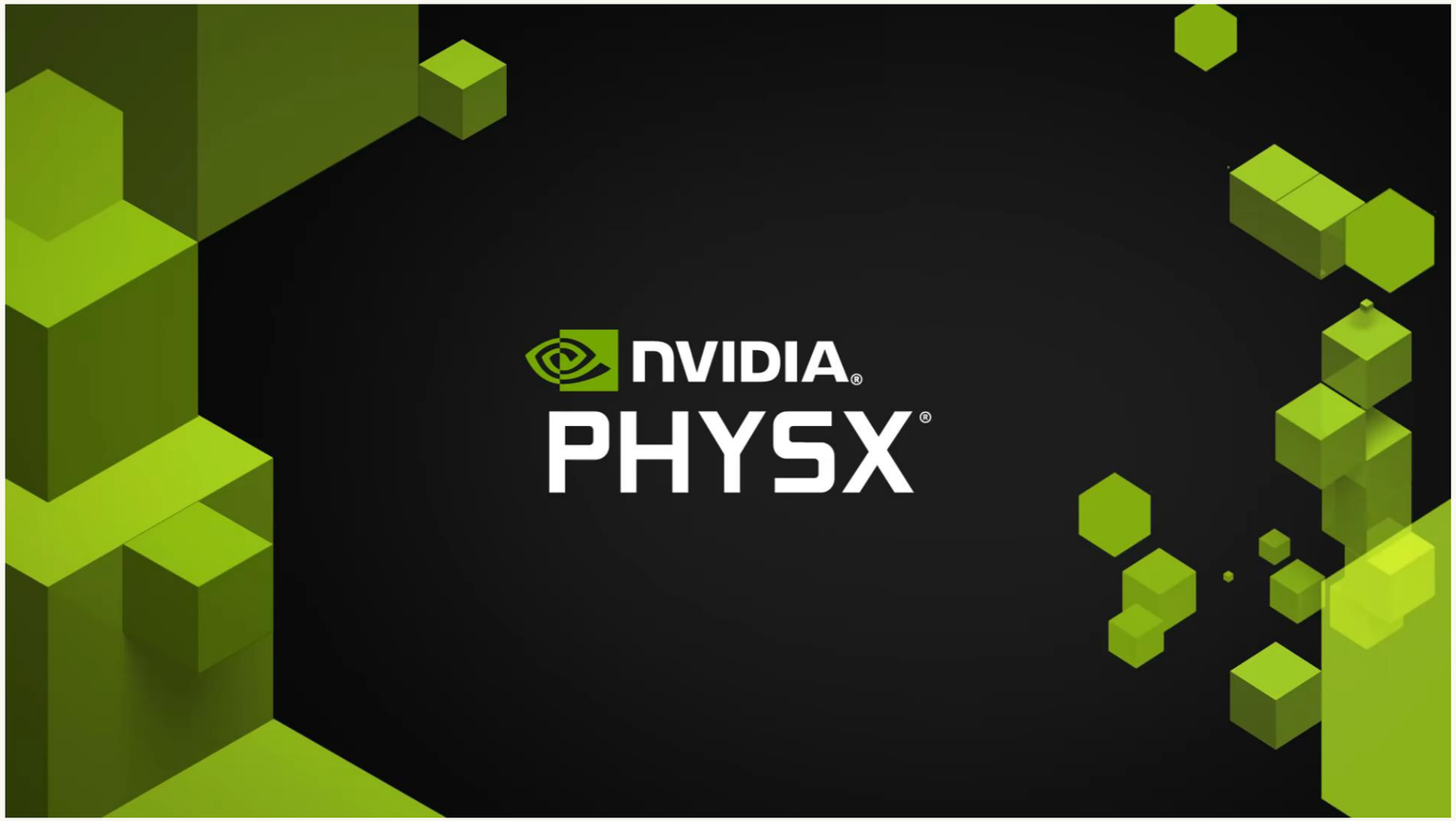
```
<physics_material id="RigidBody1-material"
name="RigidBody1-material">
 <technique_common>
    <dynamic_friction>0.200000</dynamic_friction>
    <restitution>0.600000</restitution>
    <static_friction>0.200000</static_friction>
  </technique_common>
</physics_material>
```

See Chapter 6 “COLLADA Physics Reference” in https://www.khronos.org/collada

PhysX 4

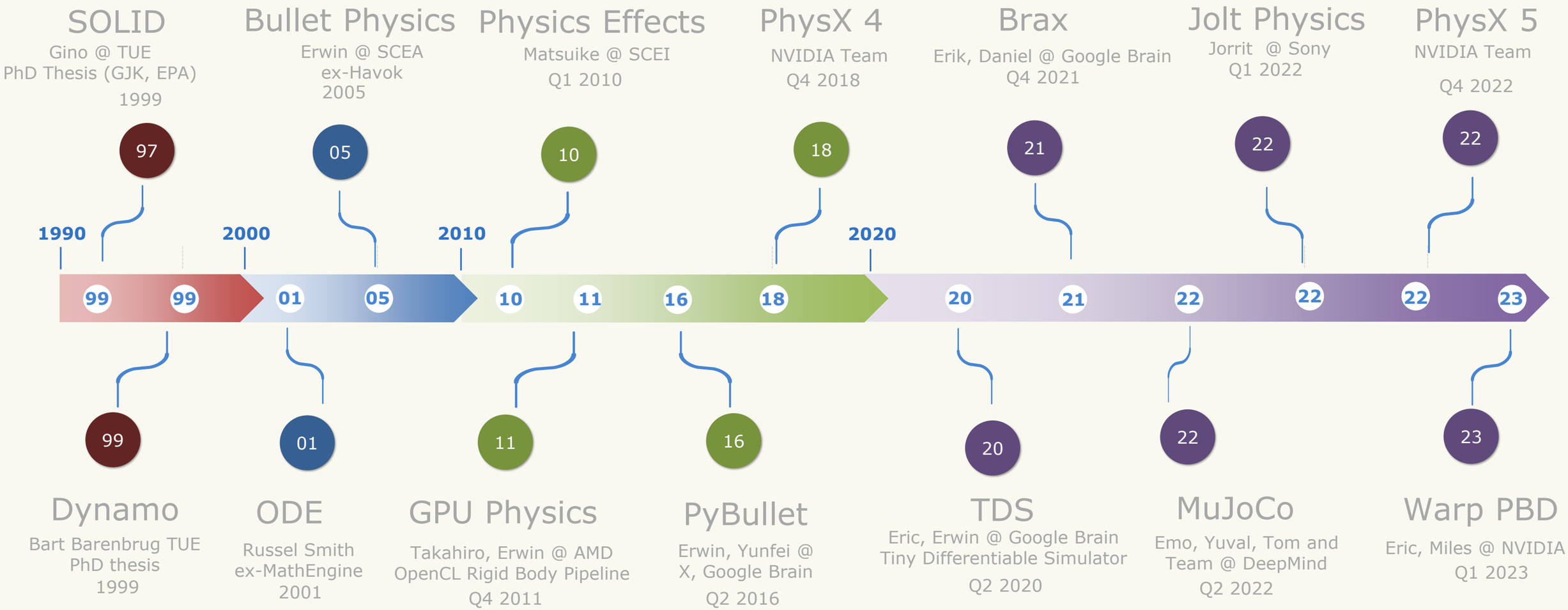

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
97
05
10
18
21
22
22
99
99
01
05
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

PhysX 4

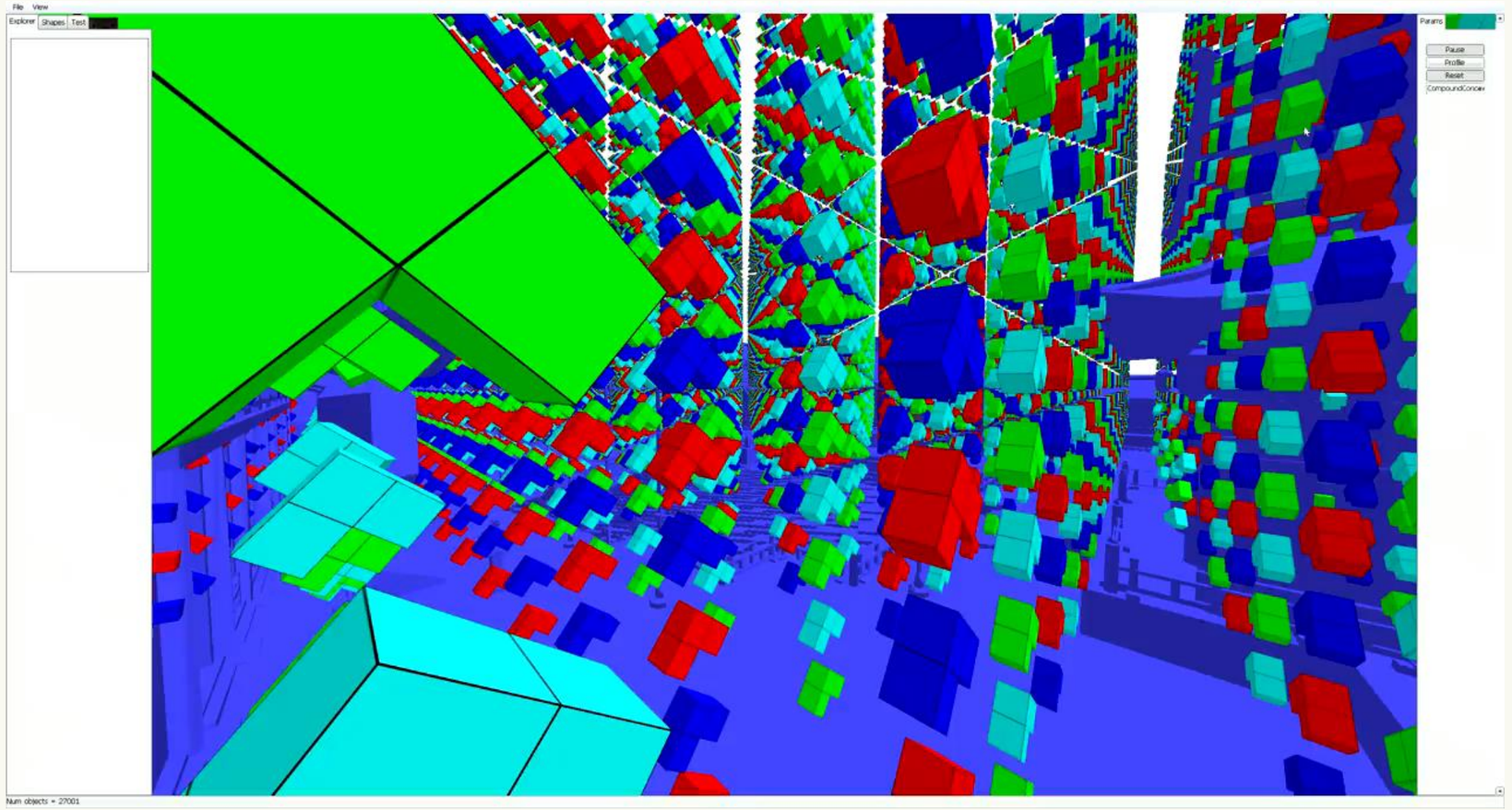


https://github.com/NVIDIAGameWorks/PhysX

GPU Physics

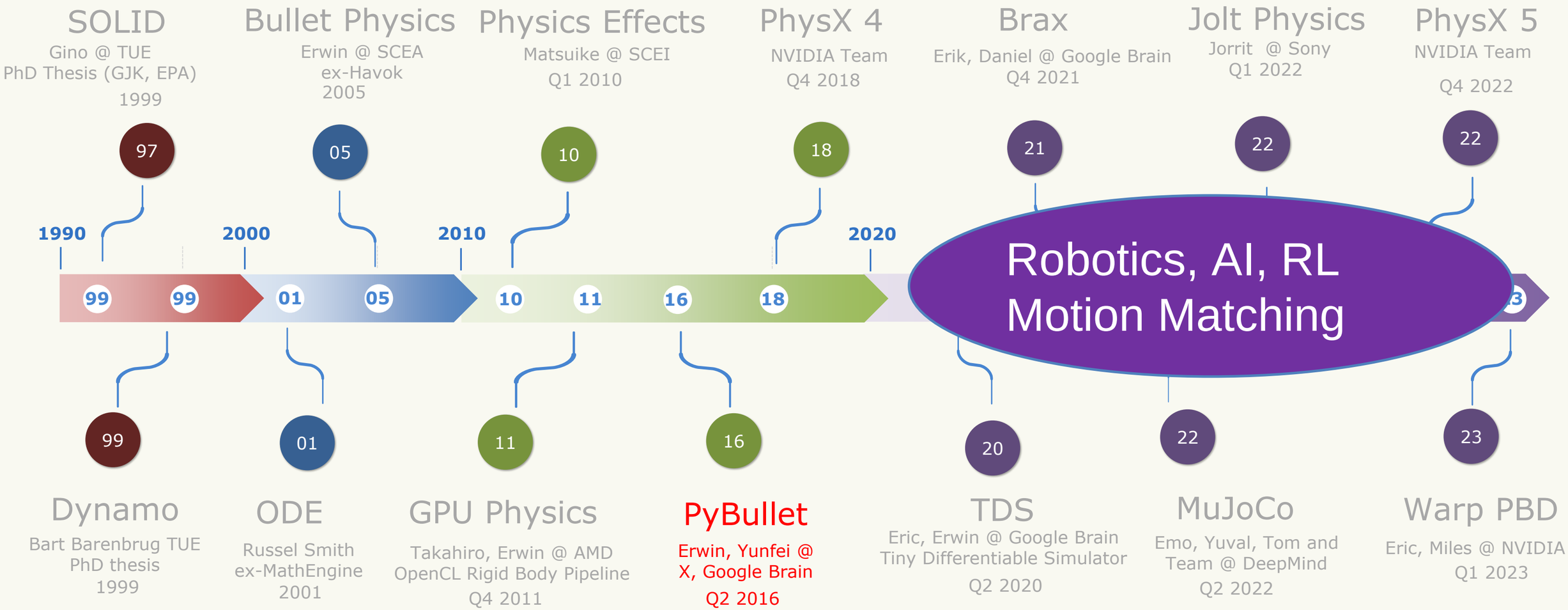

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
97
05
10
18
21
22
22
99
99
01
05
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

GPU Physics



https://github.com/bulletphysics/bullet3/tree/master/src/Bullet3OpenCL

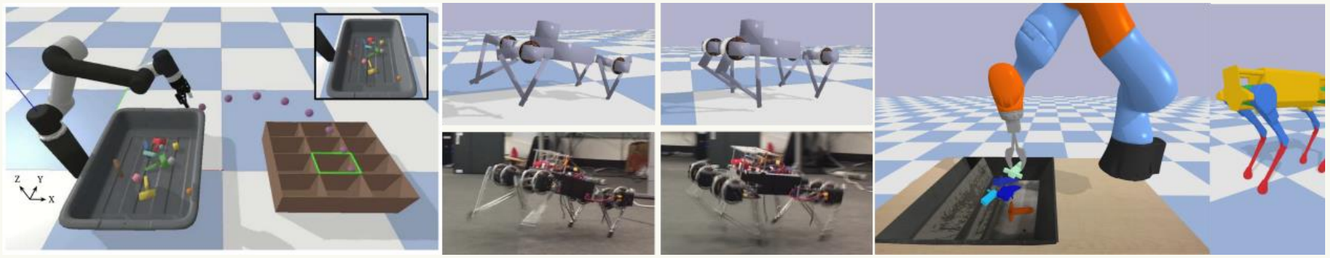
PyBullet

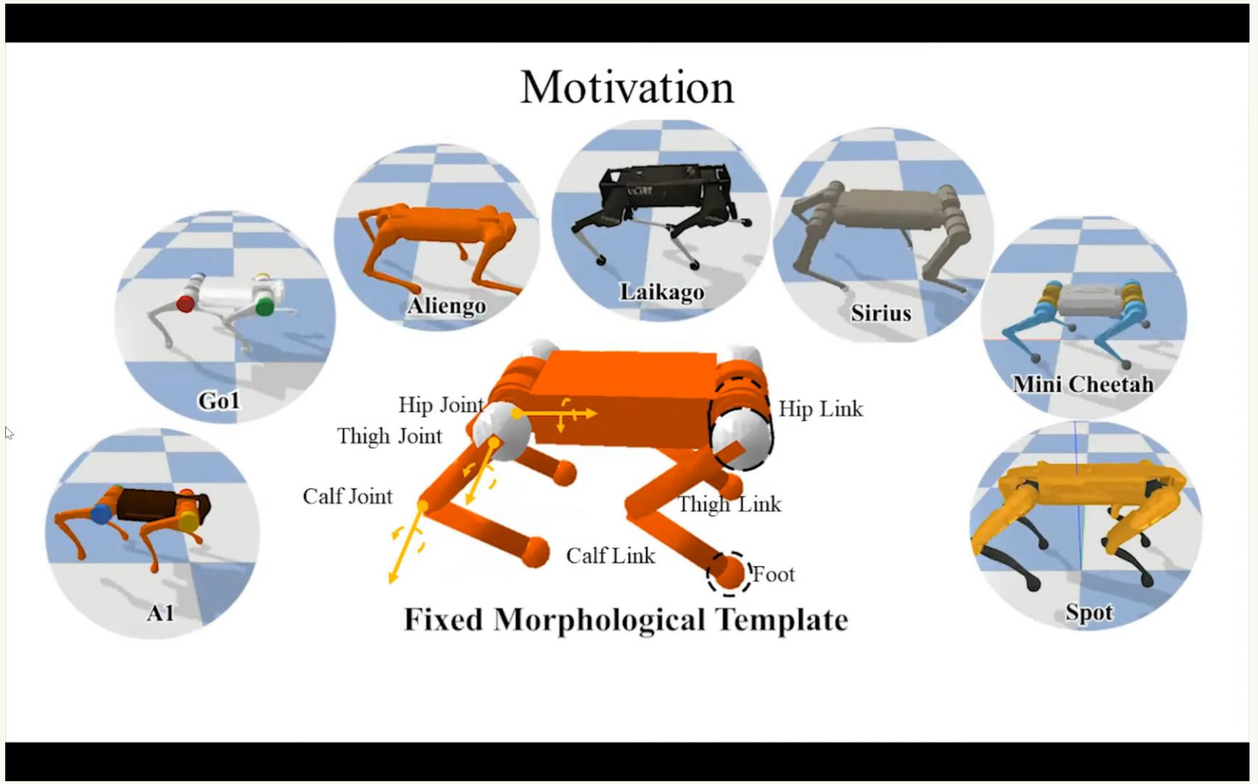

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
97
05
10
18
21
22
22
1990
2000
2010
2020
Robotics, AI, RL
Motion Matching
99
99
01
05
10
11
16
18
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

PyBullet Quickstart Guide

Erwin Coumans, Yunfei Bai, 2016-2022

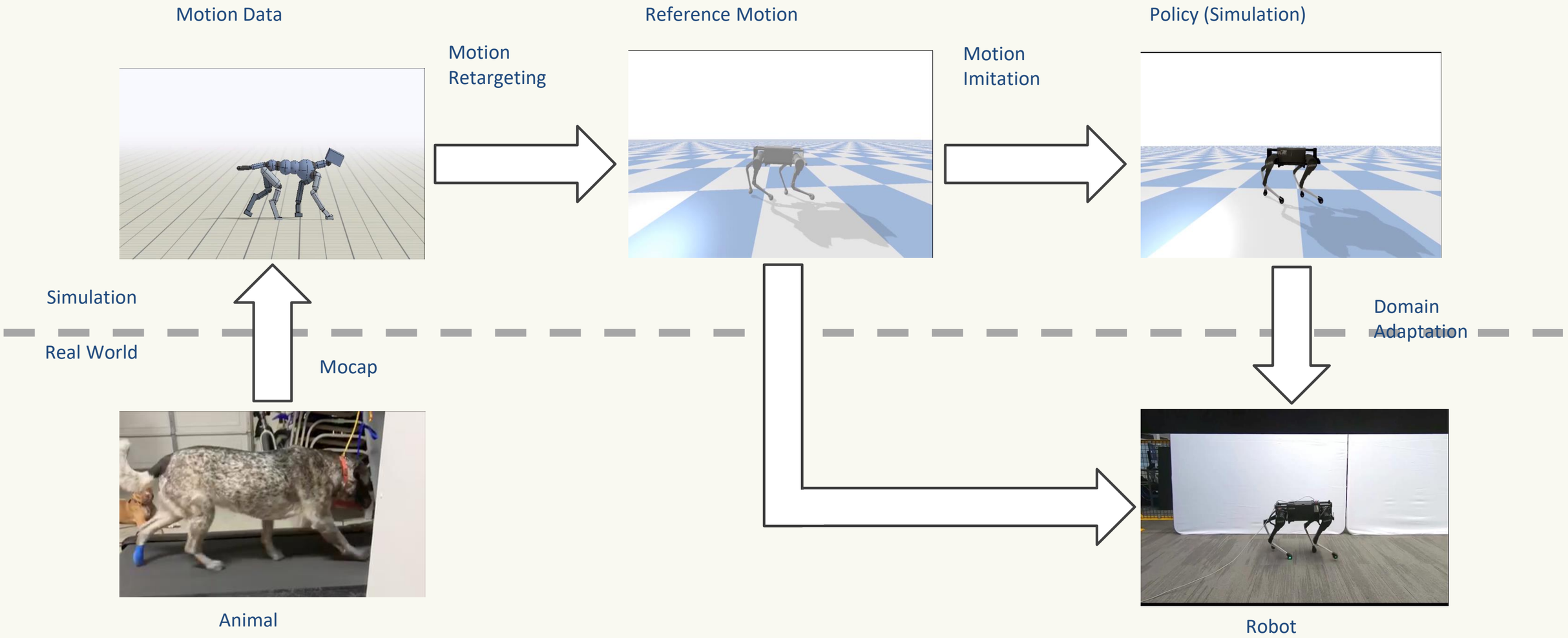
Visit desktop doc, forums, github discussions and star Bullet!





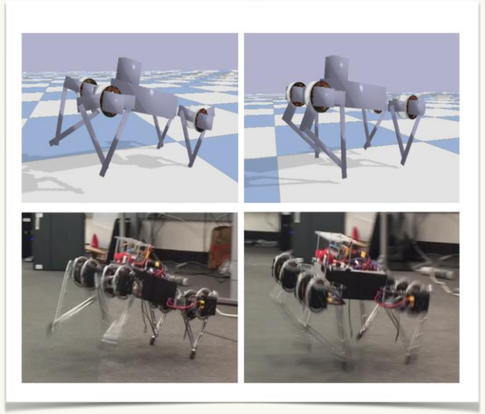
http://pybullet.org and http://wiki.ros.org/urdf/XML

Learning Agile Robotic Locomotion Skills

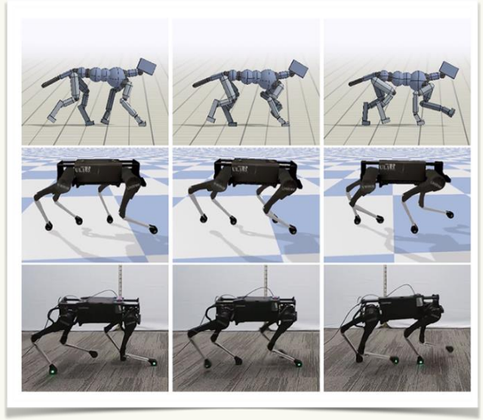


https://xbpeng.github.io/projects/Robotic_Imitation/index.html

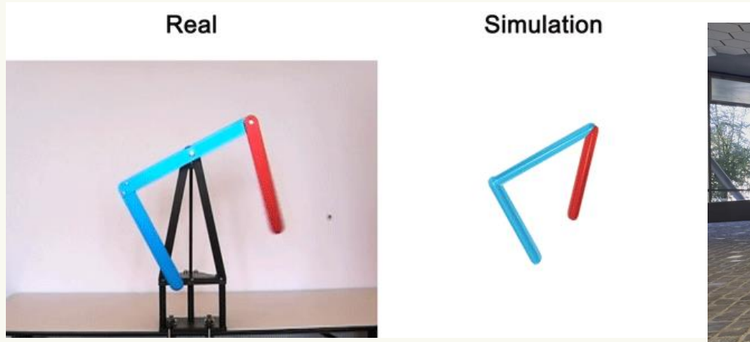
Motivation for fast Sim for Reinforcement Learning



Sim-to-Real: Learning Agile Locomotion For Quadruped Robots (2018, Jie Tan et al)

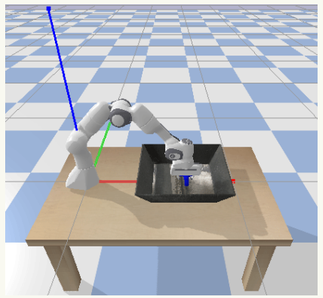


Learning Agile Robotic Locomotion Skills by Imitating Animals (2020, Jason Peng et al)

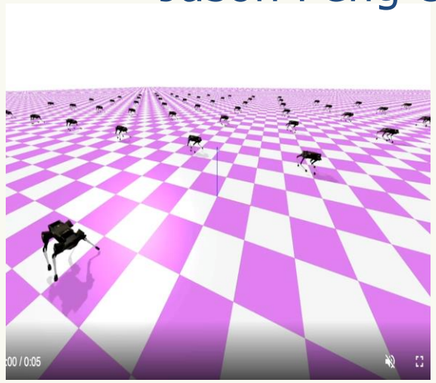


Inferring Articulated Rigid Body Dynamics from RGBD Video (2022, Eric Heiden et al)

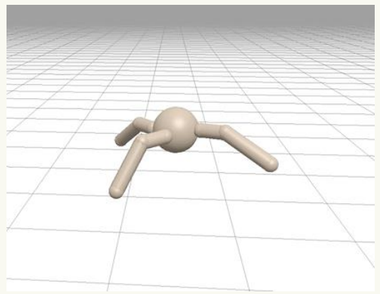
Advanced Skills through Multiple Adversarial Motion Priors in Reinforcement Learning (2022, ETH/NVIDIA)



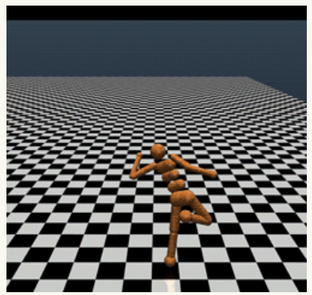
PyBullet



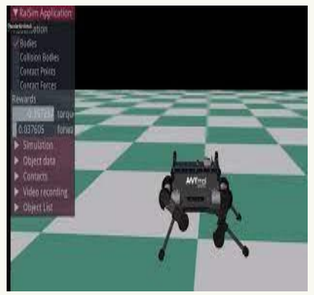
Tiny Differentiable Simulator



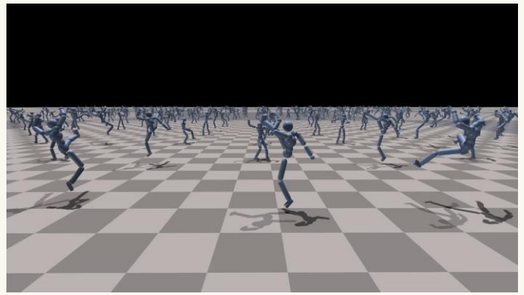
Brax



MuJoCo

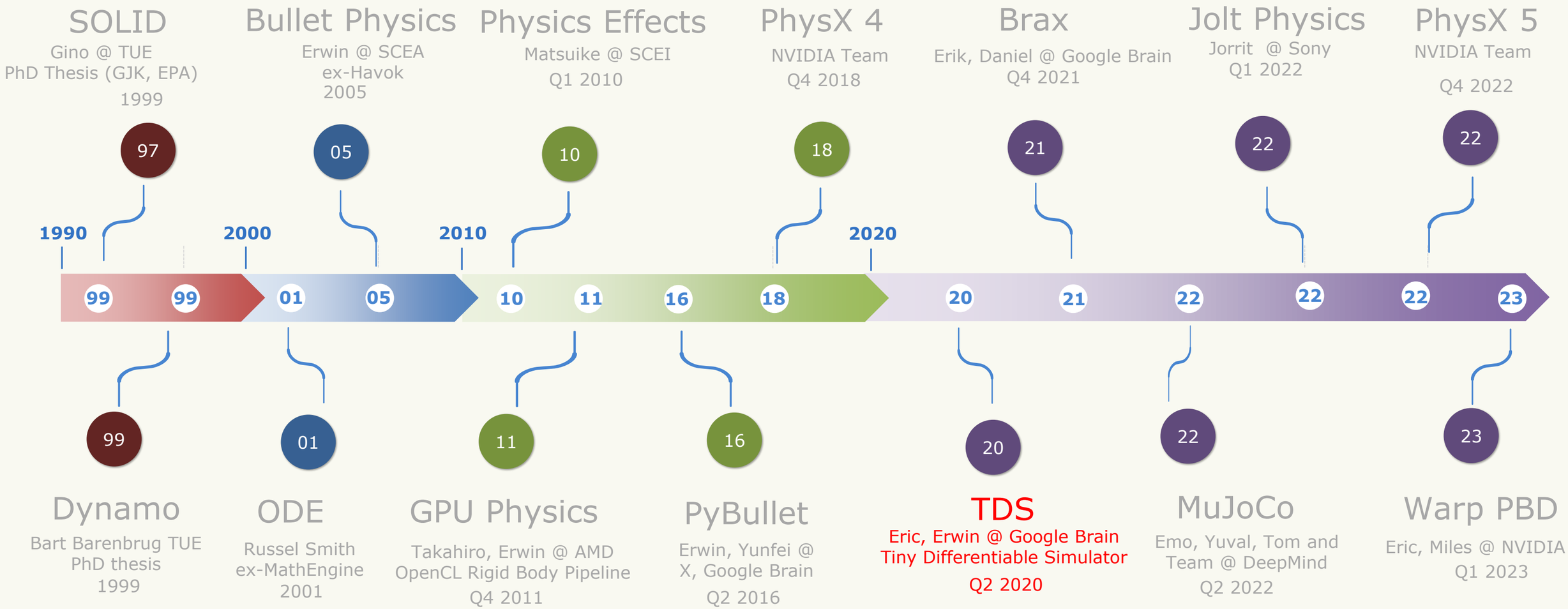


RaiSim

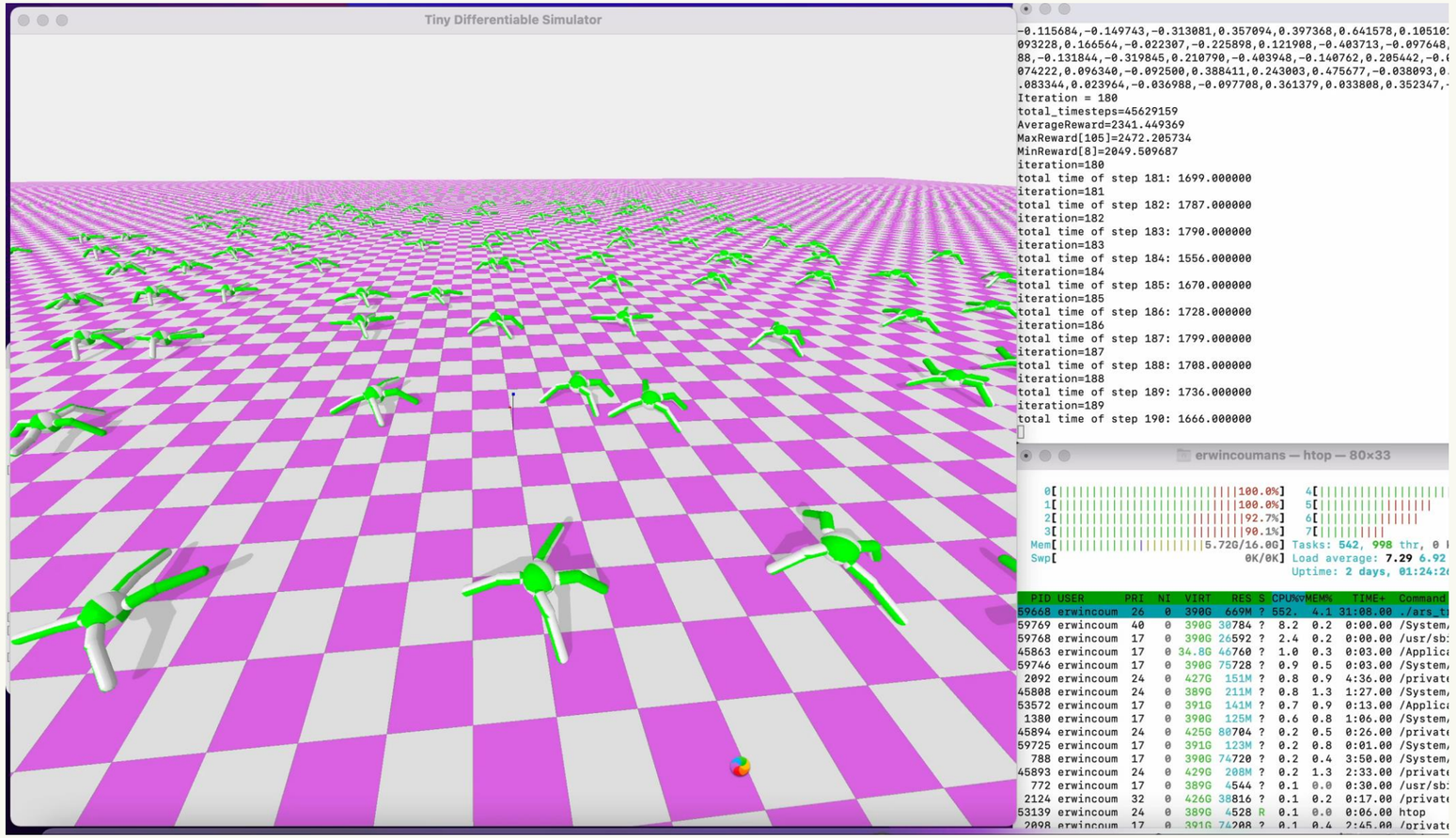


Omniverse Isaac Gym PhysX/WARP

Tiny Differentiable Simulator

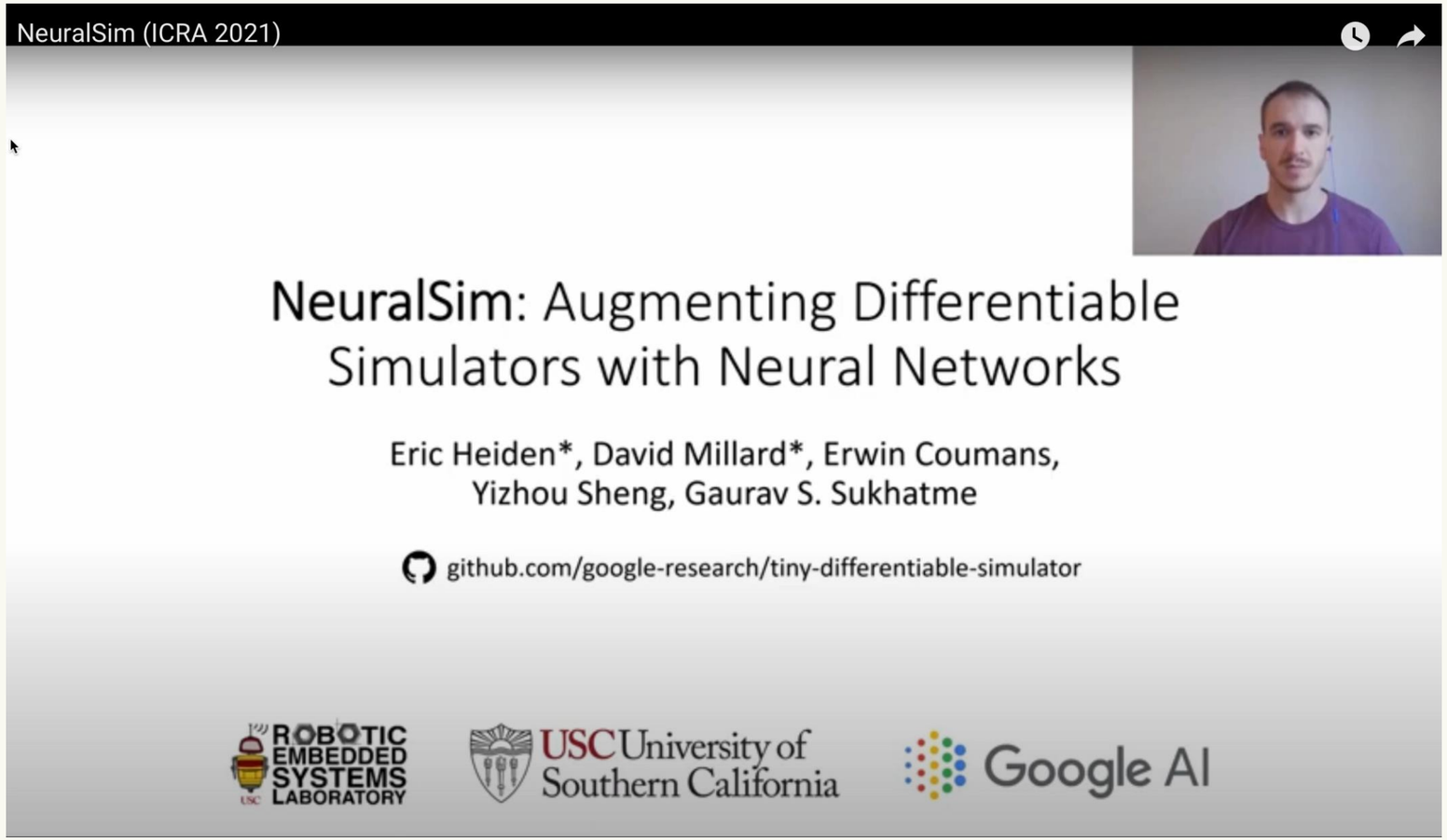


Tiny Differentiable Simulator



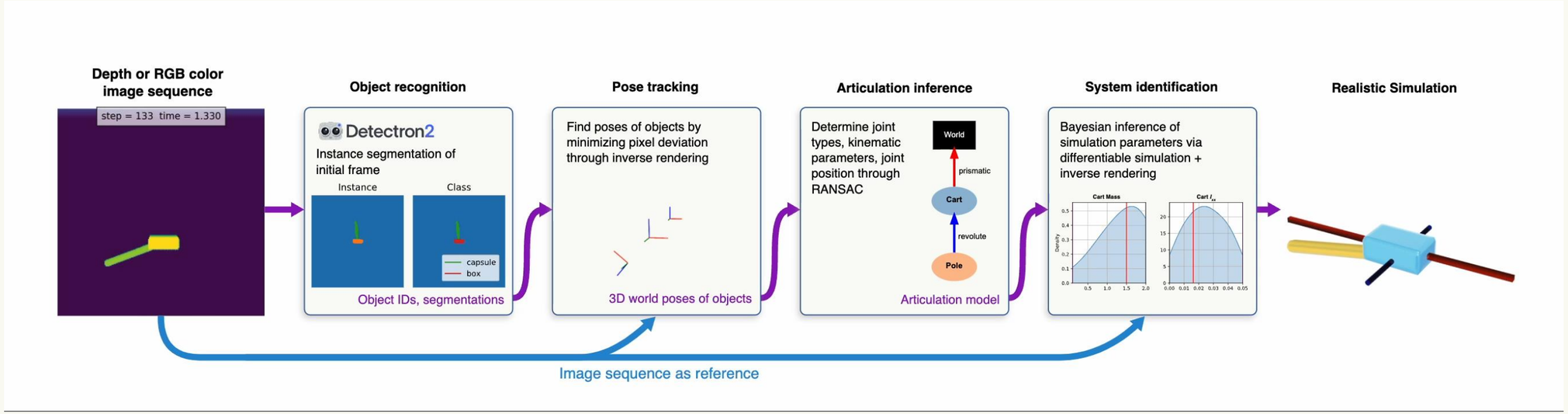
https://github.com/erwincoumans/tiny-differentiable-simulator

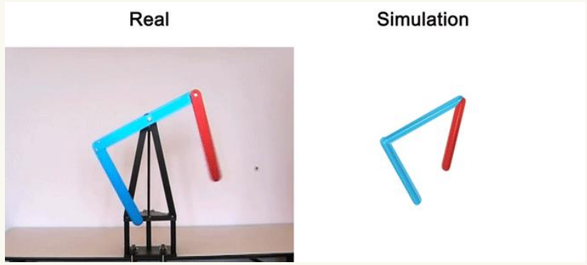
NeuralSim



https://arxiv.org/pdf/2011.04217.pdf

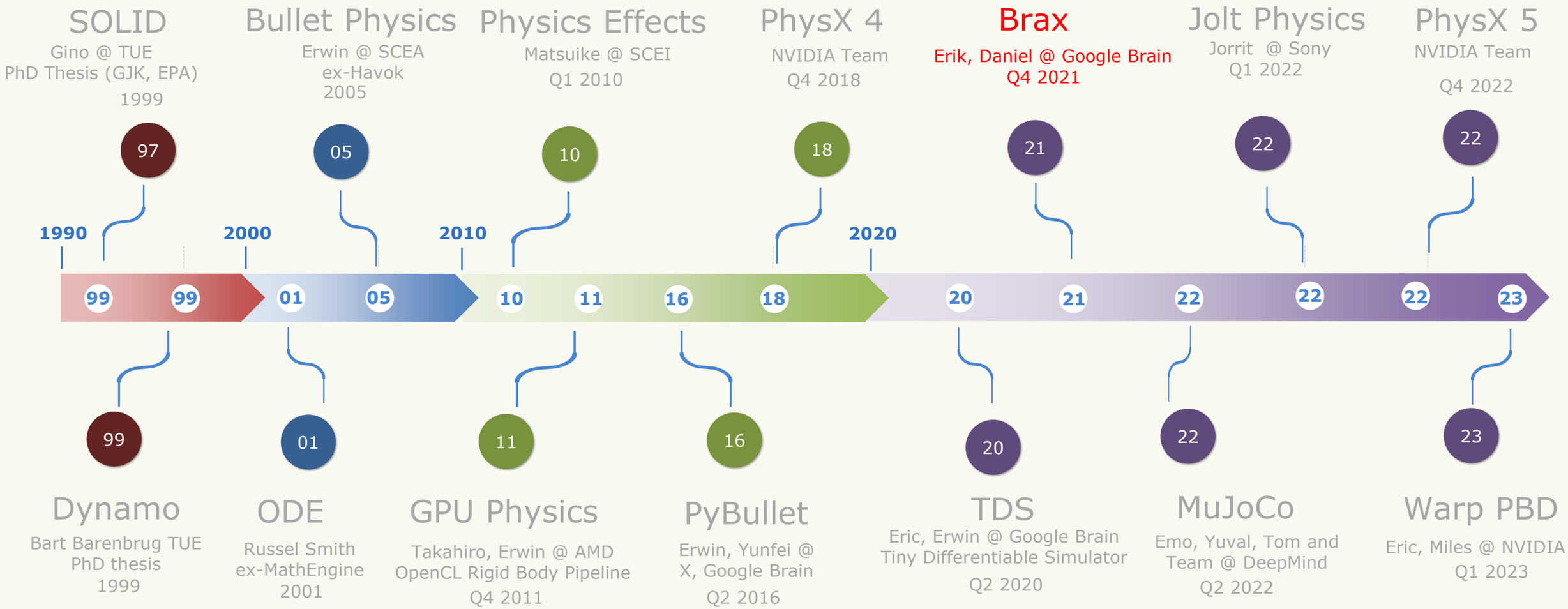
Pixels to URDF / USD



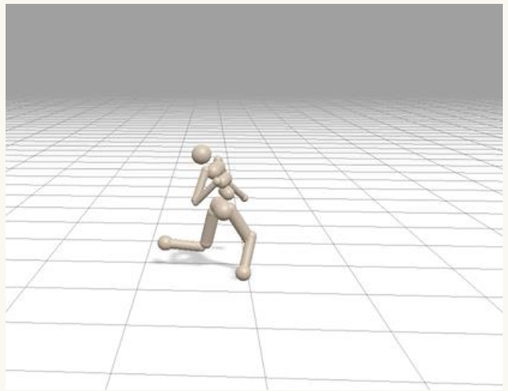


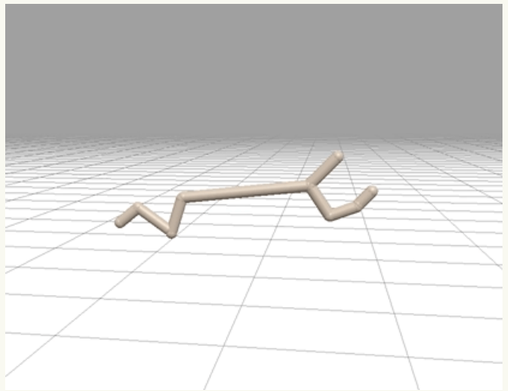
https://eric-heiden.com/publication/2022-video2sim-iros

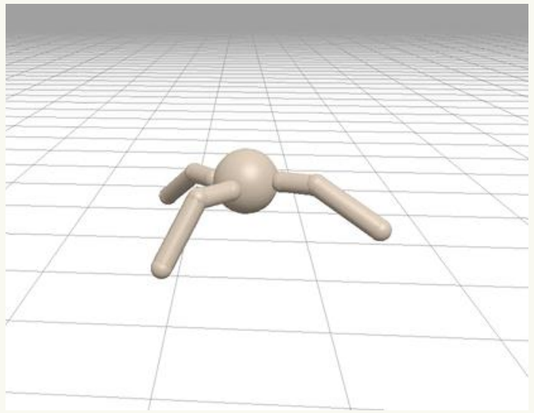
Brax

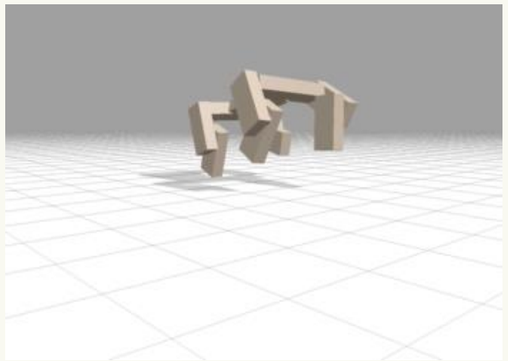

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
97
05
10
18
21
22
22
99
99
01
05
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

Brax

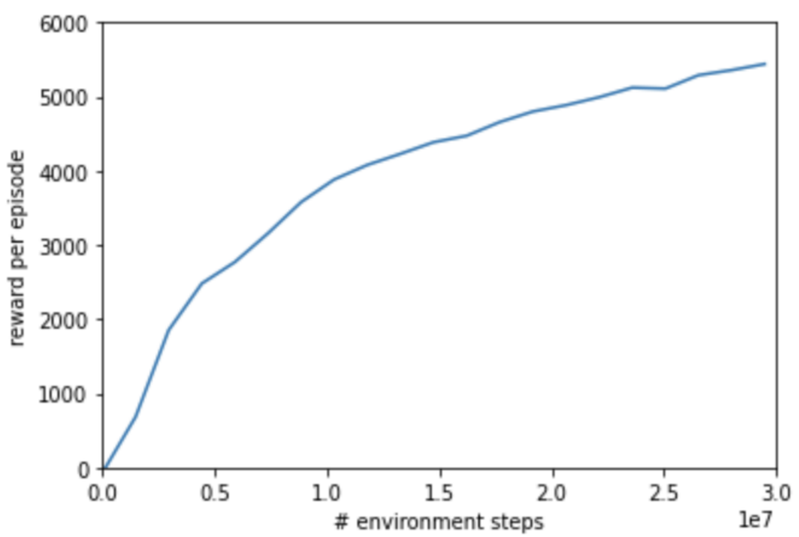








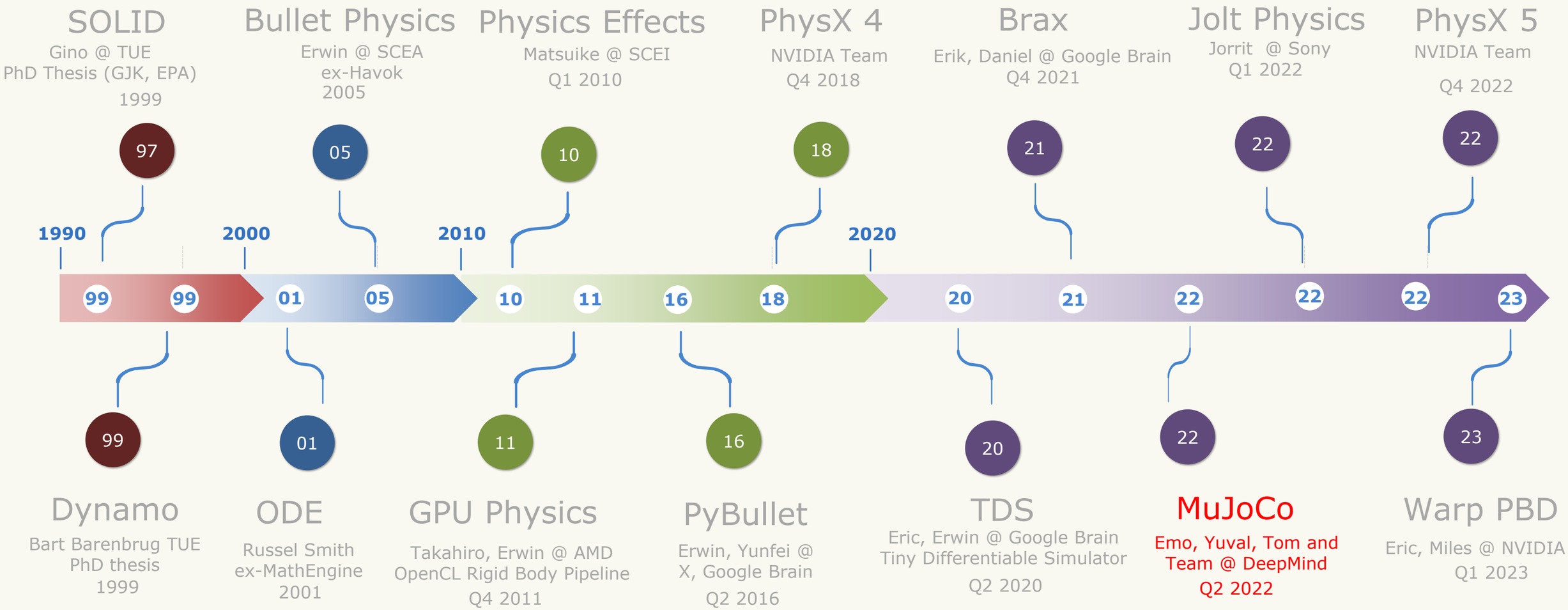
```
91
92 print(f'time to jit: {times[1] - times[0]}')
93 print(f'time to train: {times[-1] - times[1]}')
```




```
time to jit: 0:00:58.711923
time to train: 0:01:47.832345
```

https://github.com/google/brax

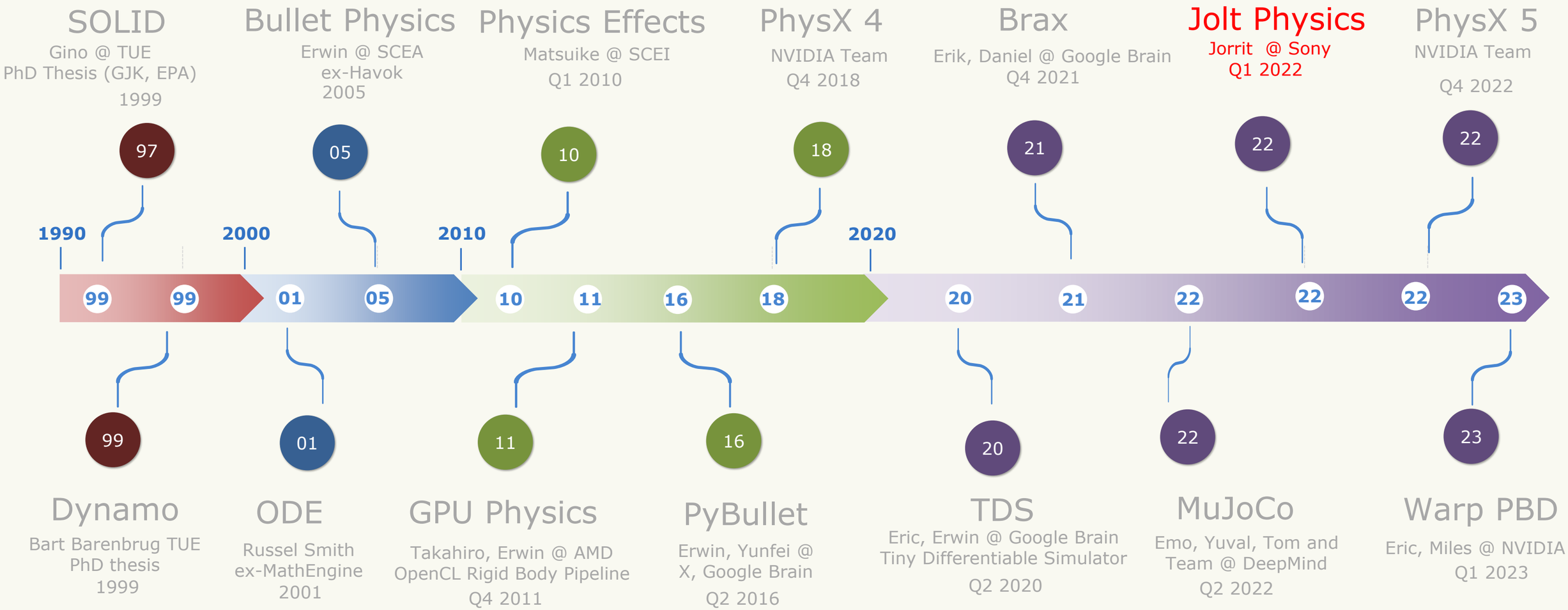
MuJoCo


SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
97
05
10
18
21
22
22
99
99
01
05
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

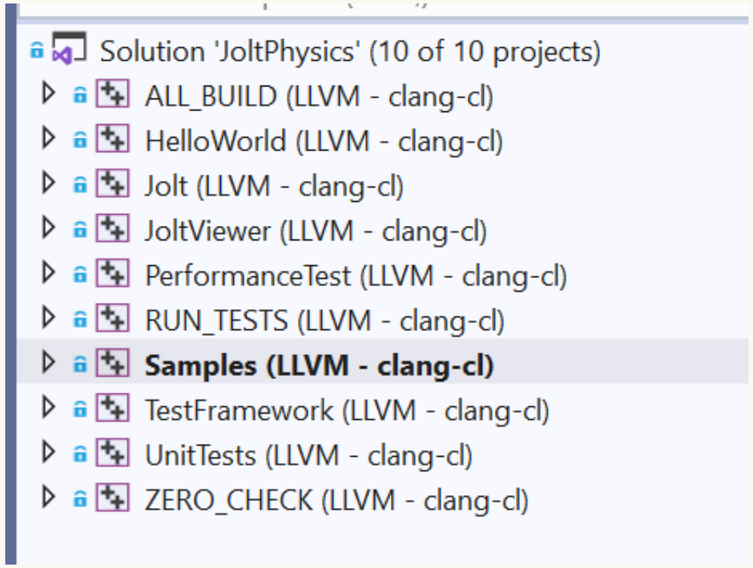
MuJoCo

https://github.com/deepmind/mujoco

Jolt Physics

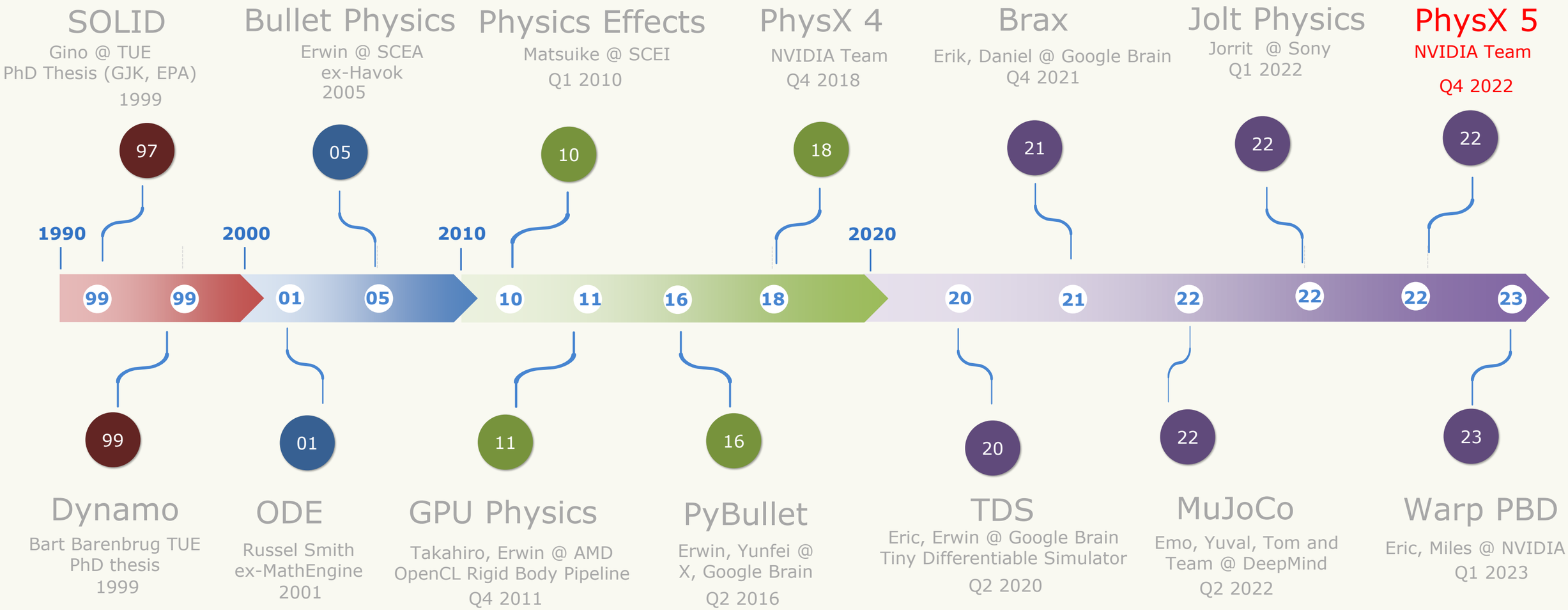


Jolt Physics

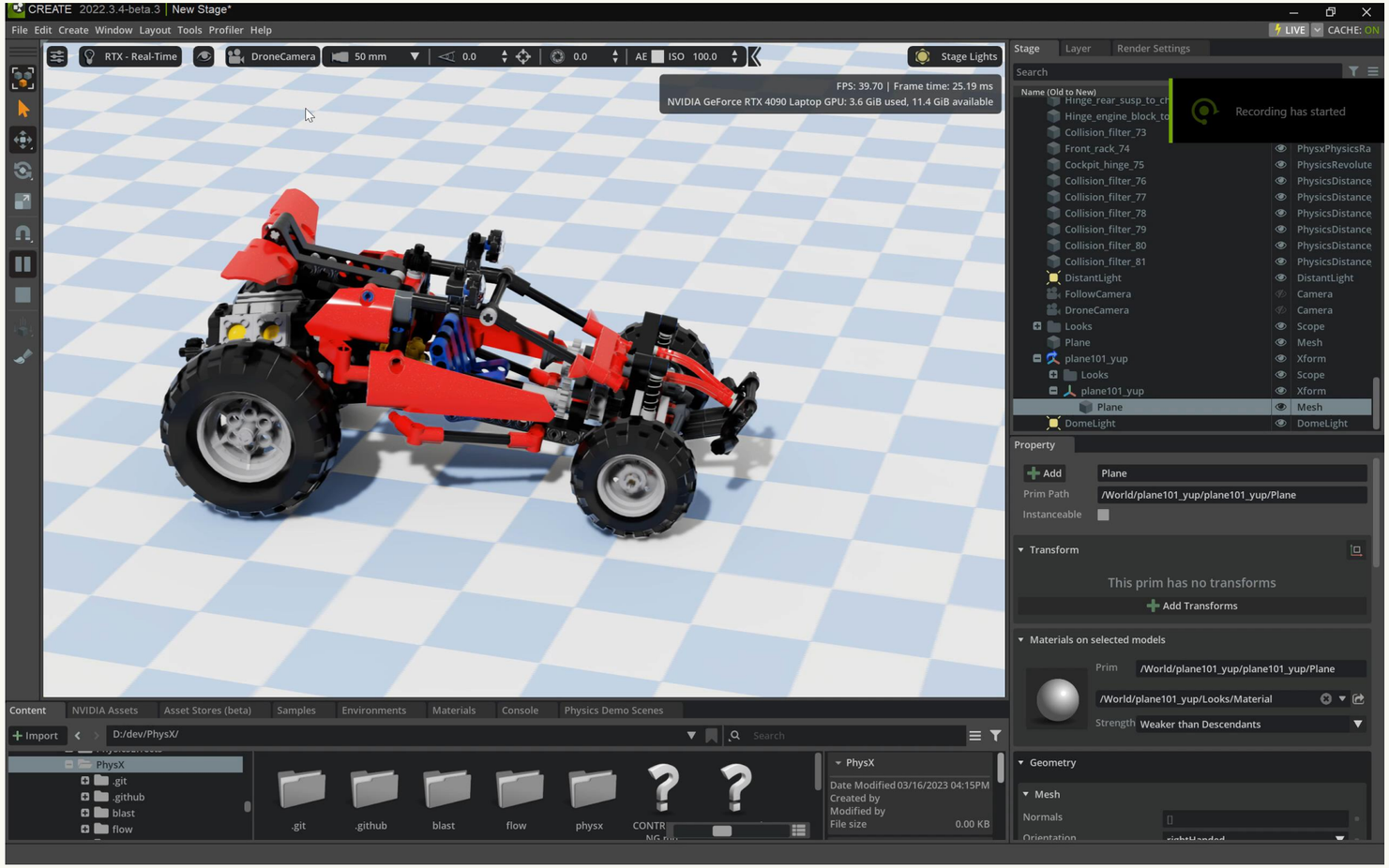


https://github.com/jrouwe/JoltPhysics

PhysX 5

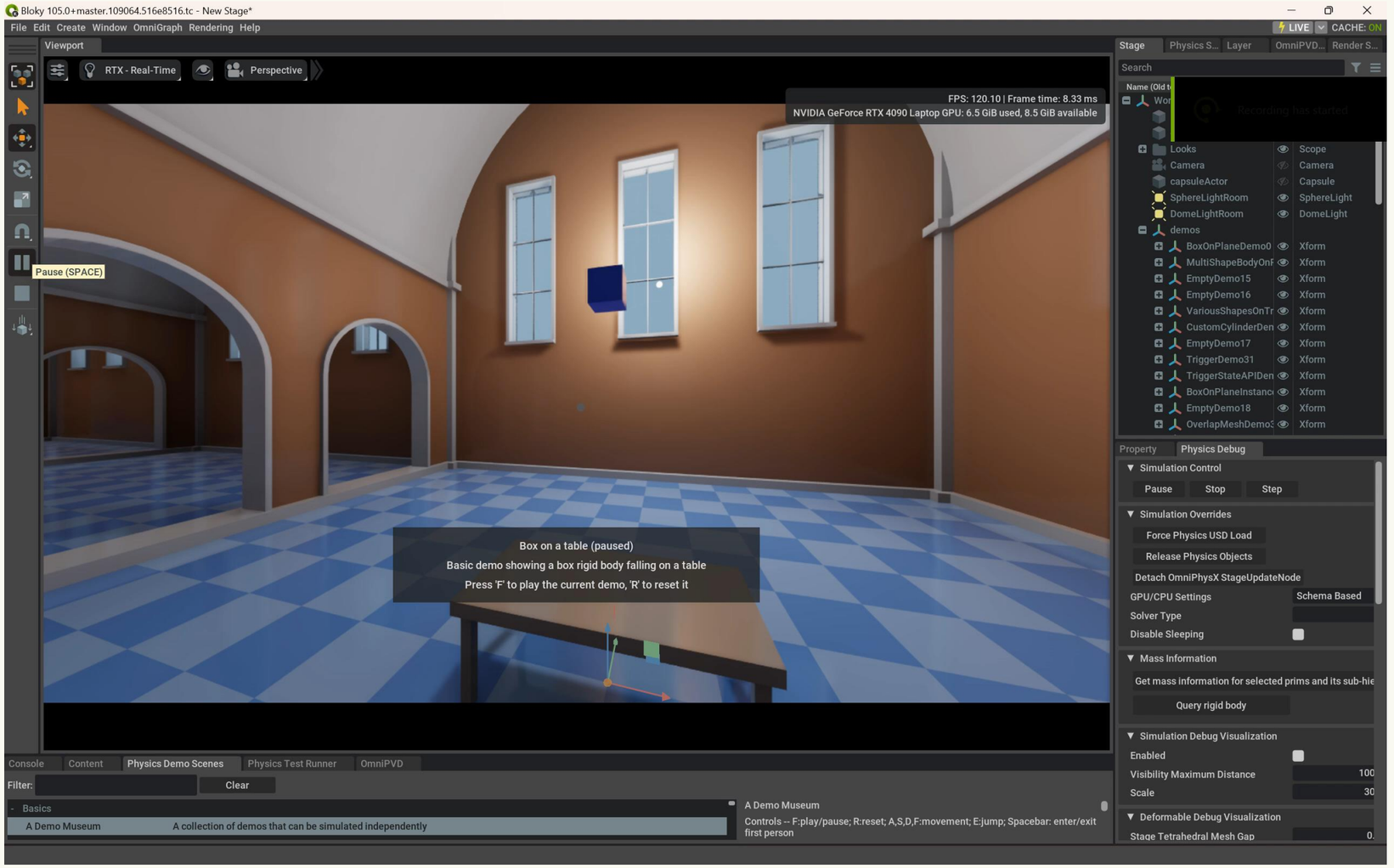

SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
97
05
10
18
21
22
22
99
99
01
05
10
11
16
18
20
21
22
22
22
23
99
01
11
16
20
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

PhysX 5 in Omniverse



https://github.com/NVIDIA-Omniverse/PhysX

PhysX 5 in Omniverse



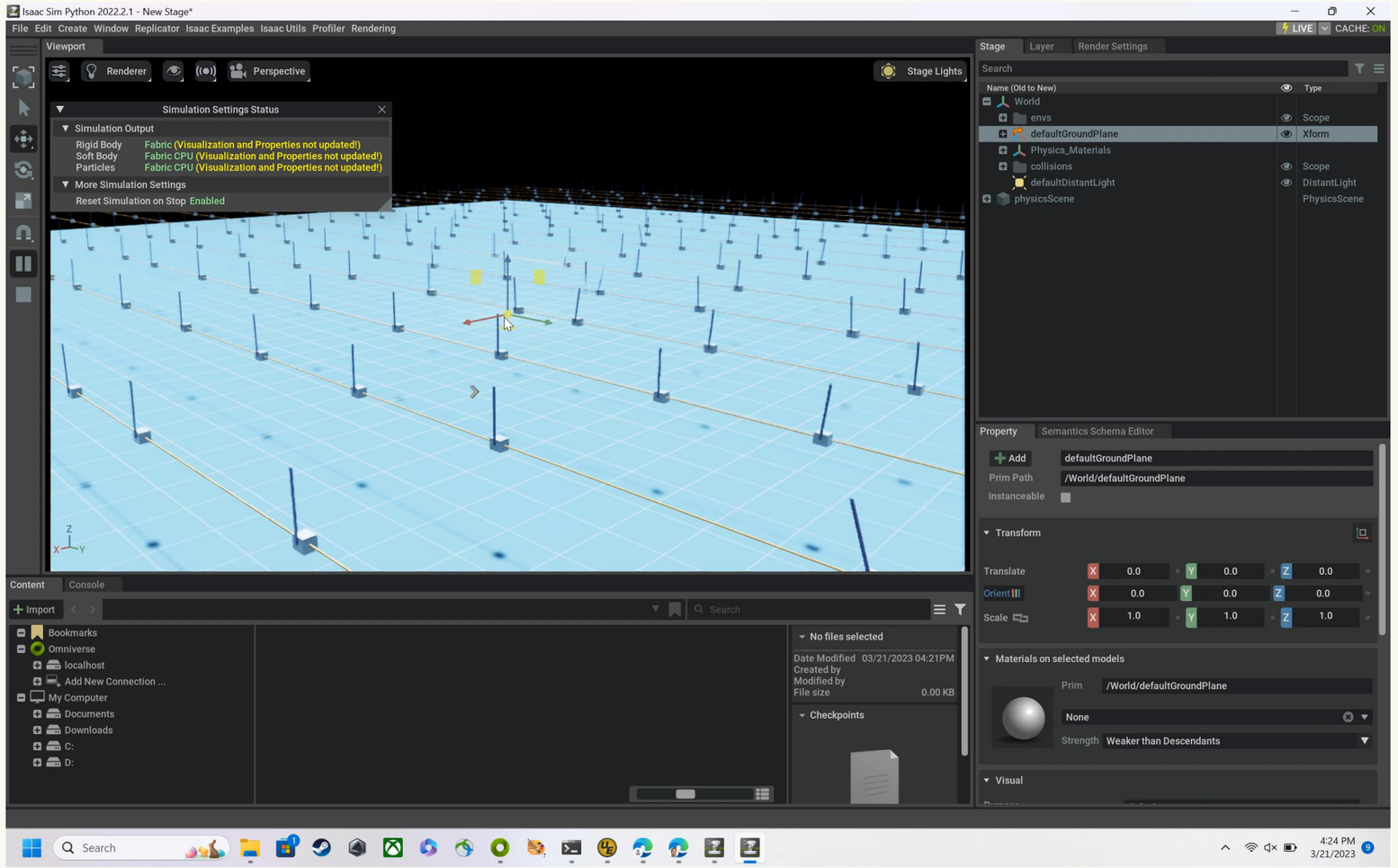
PhysX, USD, MDL in Omniverse

Open Standards

Top Industry Tools

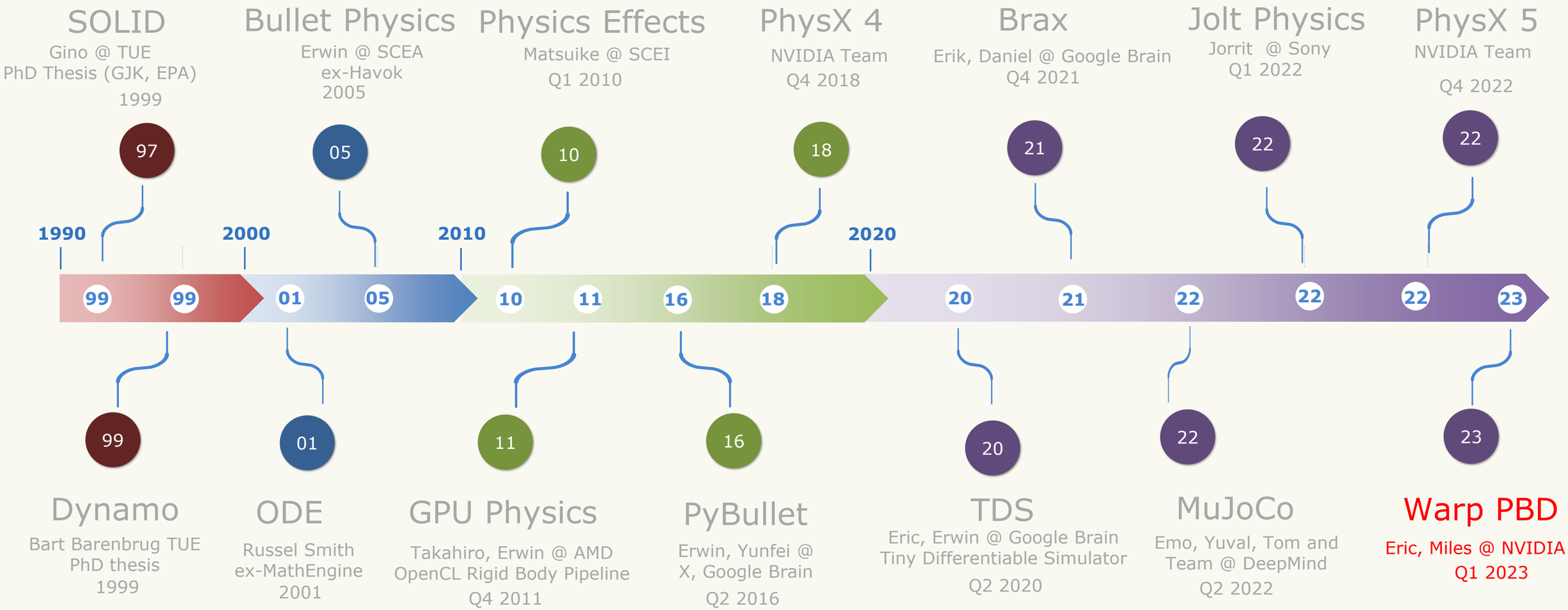
https://openusd.org/release/api/usd_physics_page_front.html

OmniIsaacGymEnvs RL training



https://github.com/NVIDIA-Omniverse/OmniIsaacGymEnvs

Warp (pbd)


SOLID
Gino @ TUE
PhD Thesis (GJK, EPA)
1999
Bullet Physics
Erwin @ SCEA
ex-Havok
2005
Physics Effects
Matsuike @ SCEI
Q1 2010
PhysX 4
NVIDIA Team
Q4 2018
Brax
Erik, Daniel @ Google Brain
Q4 2021
Jolt Physics
Jorrit @ Sony
Q1 2022
PhysX 5
NVIDIA Team
Q4 2022
1990
2000
2010
2020
99
99
01
05
10
11
16
18
20
21
22
22
22
23
Dynamo
Bart Barenbrug TUE
PhD thesis
1999
ODE
Russel Smith
ex-MathEngine
2001
GPU Physics
Takahiro, Erwin @ AMD
OpenCL Rigid Body Pipeline
Q4 2011
PyBullet
Erwin, Yunfei @
X, Google Brain
Q2 2016
TDS
Eric, Erwin @ Google Brain
Tiny Differentiable Simulator
Q2 2020
MuJoCo
Emo, Yuval, Tom and
Team @ DeepMind
Q2 2022
Warp PBD
Eric, Miles @ NVIDIA
Q1 2023

WARP

Python-based CUDA Kernel JIT – “Shadertoy for simulation”

Warp Goals

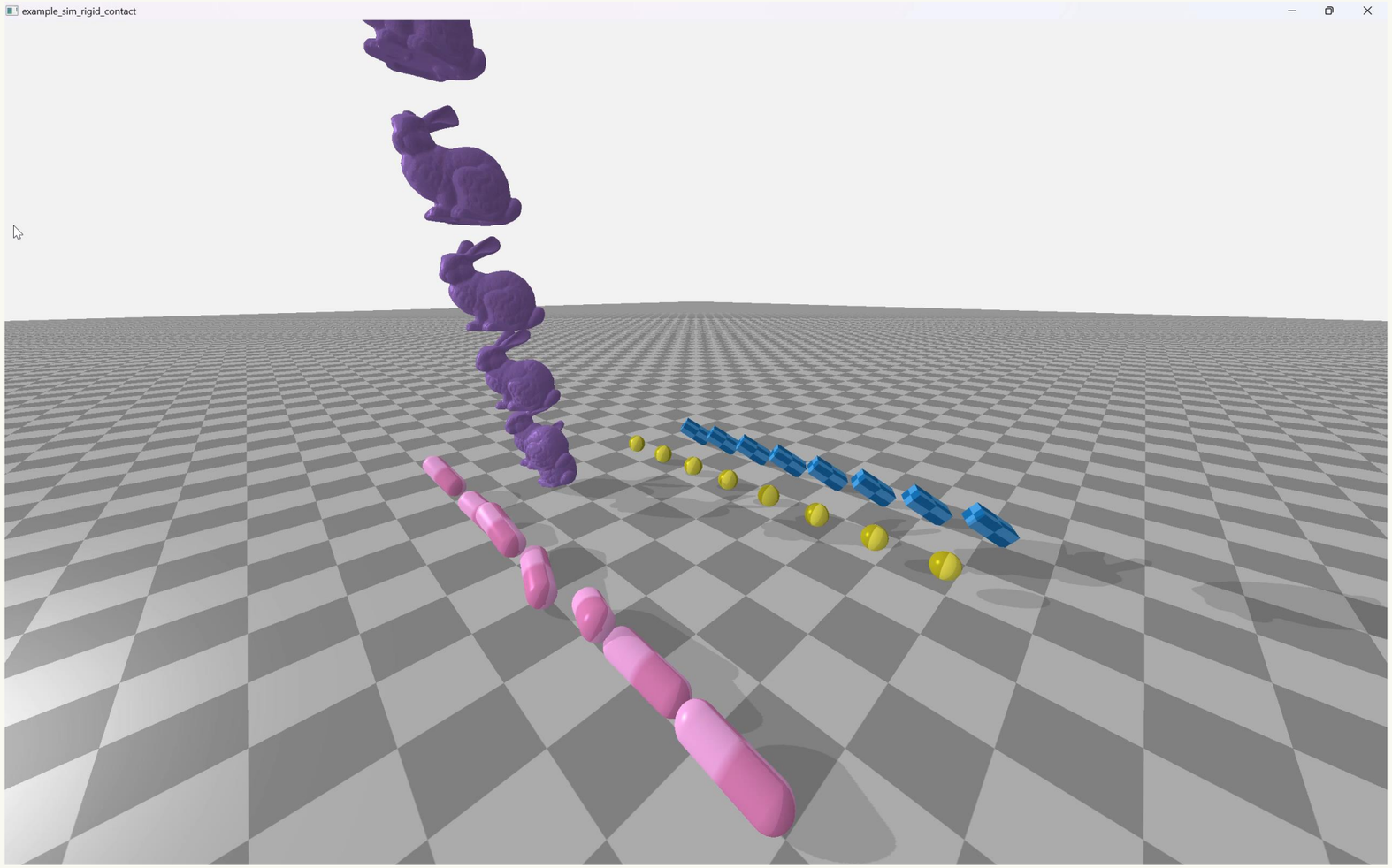
1. Make it easy to write GPU graphics + simulation code from Python
2. Lower barrier to entry, faster iteration time
3. Rich built-in math, geometry library
4. Simple interop with existing C++/CUDA code
5. Fully differentiable

```python
import warp as wp

@wp.kernel
def calculate_reward_wp(obs_buf: wp.array(dtype=float),
                        rew_buf: wp.array(dtype=float),
                        reset_buf: wp.array(dtype=int),
                        timeout_buf: wp.array(dtype=int),
                        progress_buf: wp.array(dtype=int),
                        termination_height: float,
                        num_obs: int,
                        max_episode_length: int):

    tid = wp.tid()

    timeout_buf[tid] = 0
    reset_buf[tid] = 0
    progress_buf[tid] = progress_buf[tid] + 1

    up_reward = 0.1 * obs_buf[tid * num_obs + 27]
    heading_reward = obs_buf[tid * num_obs + 28]
    height_reward = obs_buf[tid * num_obs] - termination_height
    progress_reward = obs_buf[tid * num_obs + 5]

    rew_buf[tid] = progress_reward + up_reward + heading_reward + height_reward

    # reset agents
    if progress_buf[tid] >= max_episode_length:
        reset_buf[tid] = 1
        timeout_buf[tid] = 1

    if height_reward < 0.0:
        reset_buf[tid] = 1

    if reset_buf[tid] == 1:
        progress_buf[tid] = 0
```

Ant RL Reward Function in Warp – torch.where() begone!

Warp: A High-performance Python Framework for GPU Simulation and Graphics

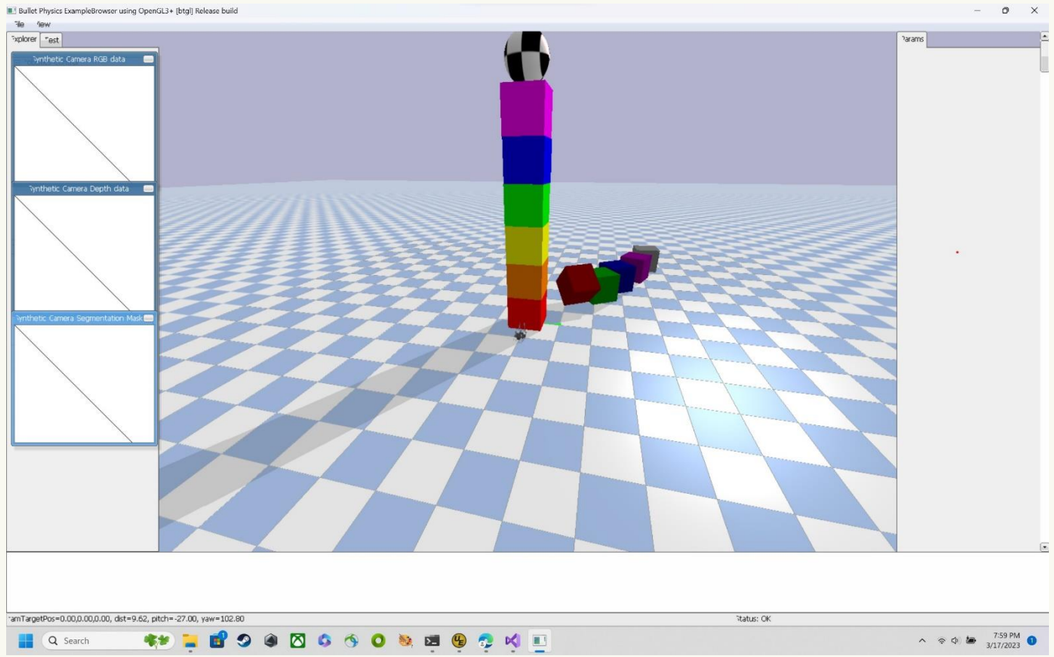
Warp



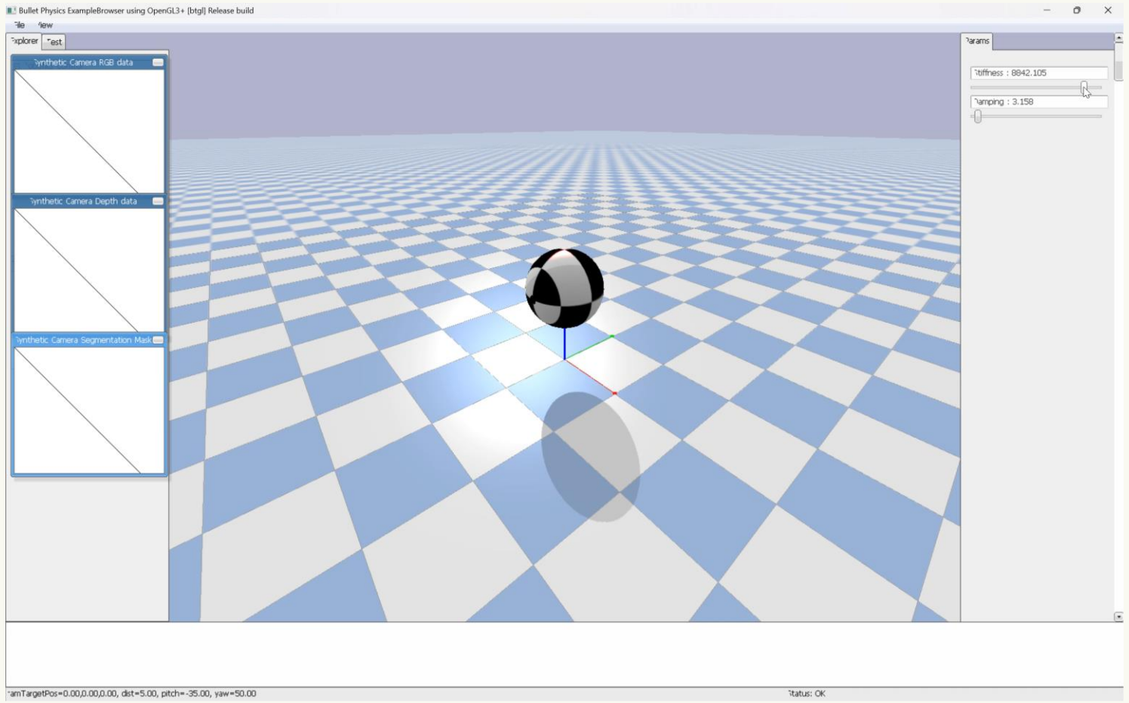
https://github.com/NVIDIA/warp (and https://github.com/eric-heiden/warp)

ChatGPT

"can you create a pybullet script with a ground plane, a box and a sphere on top."+"can you add 10 more boxes on top?"+"can you move the 5th box 0.3 units along the x axis?"+"can you add a quadruped robot next to the boxes?"

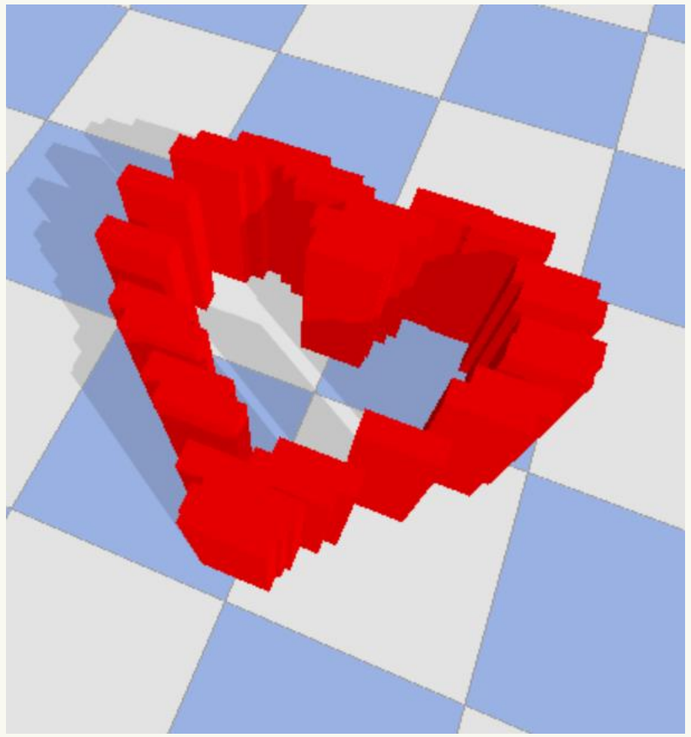


“create a pybullet script with a sphere falling on a ground plane. Add a slider to control contactStiffness and contactDamping”



ChatGPT PyBullet

“Place 100 dominoes in a shape of a heart”



```python
import pybullet as p
import time
import math

# Define the size and position of the dominoes.
domino_width = 0.3
domino_height = 1.5
domino_depth = 0.1
spacing = 0.05
domino_mass = 0.
domino_friction = 0.5
domino_restitution = 0.1
start_pos = [-4, 0, 0]

# Create a function to add the dominoes to the simulation.
def add_domino(x, y, z):
    boxId = p.createCollisionShape(p.GEOM_BOX, halfExtents=[domino_width/2, domino_depth/2, domino_height/2])
    boxVisualId = p.createVisualShape(p.GEOM_BOX, halfExtents=[domino_width/2, domino_depth/2, domino_height/2], rgbaColor=[1,0,0,1])
    dominoId = p.createMultiBody(domino_mass, boxId, boxVisualId, [x, y, z])
    p.changeDynamics(dominoId, -1, lateralFriction=domino_friction, restitution=domino_restitution)
    return dominoId

# Define the positions of the dominoes in the shape of a heart.
heart_pos = []
for t in range(0, 360, 10):
    x = 16 * math.sin(t) ** 3
    y = -13 * math.cos(t) + 5 * math.cos(2*t) + 2 * math.cos(3*t) + math.cos(4*t)
    heart_pos.append([start_pos[0] + x * spacing, start_pos[1] + y * spacing, start_pos[2]])

# Create a simulation environment and set the gravity.
p.connect(p.GUI)
p.setGravity(0, 0, -10)

# Create a ground plane.
planeId = p.createCollisionShape(p.GEOM_PLANE)
p.createMultiBody(0, planeId)

# Add the dominoes to the simulation.
for pos in heart_pos:
    add_domino(pos[0], pos[1], pos[2])

# Run the simulation.
for i in range(10000):
    p.stepSimulation()
    time.sleep(1./240.)

# Disconnect the simulation environment.
p.disconnect()
```

GDC | March 20-24, 2023 San Francisco, CA

Thank You! Questions?

https://twitter.com/erwincoumans

Don’t forget about the speaker survey.

#GDC23