Programming Vuforia in Unity: some advices

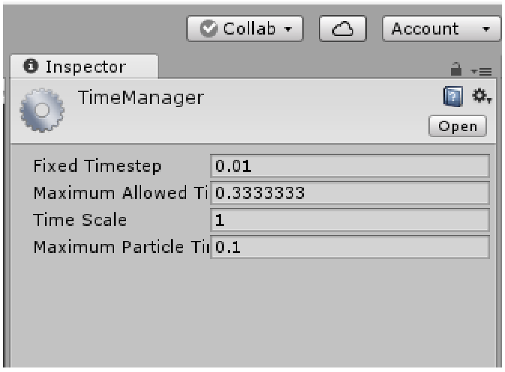
J. Vezien – Nov 2022

Programming a game in Unity is relatively easy. Using Vuforia is also ok, if you are a bit careful. But setting up a full AR game combining Vuforia with the Unity physics engine is sometimes frustrating. Here are some tips:

- Set the scale properly: Vuforia markers will be in centimeters, and will appear at their true scale in Unity, where the unit is… the meter by default. So make your models at the proper scale, or downscale them by the correct amount if they are tracked objects (that is, daughters of a Vuforia marker)

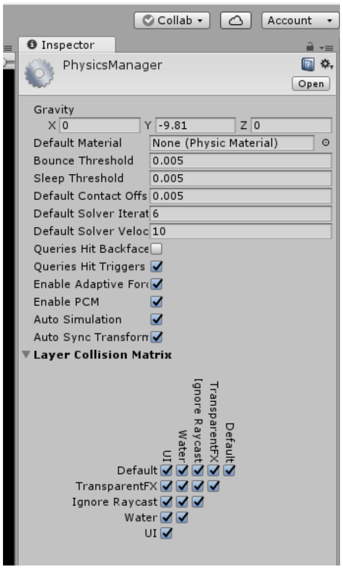
This implies that the physics engine must work at small scales… and it doesn't by default! So you need to:

- Set the time factors (in "Settings→ Time" menu):



Fixed Timestep	0.01
Maximum Allowed Ti	0.3333333
Time Scale	1
Maximum Particle Ti	0.1

- Set the physics settings ((in "Settings→ Physics" menu), in particular thresholds may be too high by default, so objects may never really "stop":



PhysicsManager

Gravity X 0 Y -9.81 Z 0

Default Material None (Physic Material)

Bounce Threshold 0.005

Sleep Threshold 0.005

Default Contact Offs 0.005

Default Solver Iterat 6

Default Solver Veloc 10

Queries Hit Backface ☐

Queries Hit Triggers ☑

Enable Adaptive Forc ☑

Enable PCM ☑

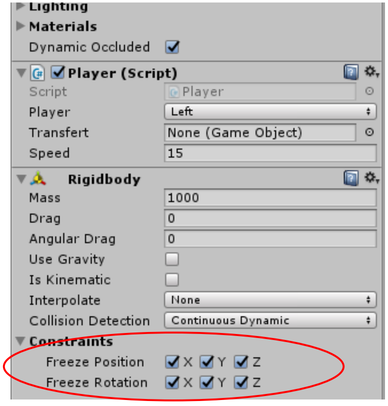
Auto Simulation ☑

Auto Sync Transform ☑

Layer Collision Matrix

	UI	Water	Ignore Raycast	TransparentFX	Default
Default	☑	☑	☑	☑	☑
TransparentFX	☑	☑	☑	☑	
Ignore Raycast	☑	☑	☑		
Water	☑	☑			
UI	☑				

- Give your objects the right properties:
  - Objects should be fixed relative to their respective markers:



Lighting

Materials

Dynamic Occluded ☑

☑ Player (Script)

Script Player

Player Left

Transfert None (Game Object)

Speed 15

Rigidbody

Mass 1000

Drag 0

Angular Drag 0

Use Gravity ☐

Is Kinematic ☐

Interpolate None

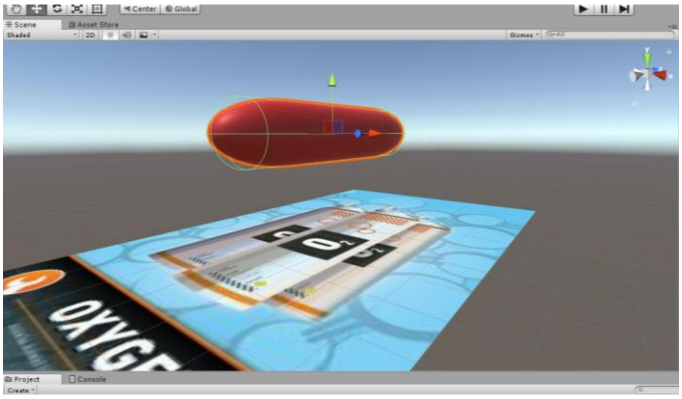
Collision Detection Continuous Dynamic

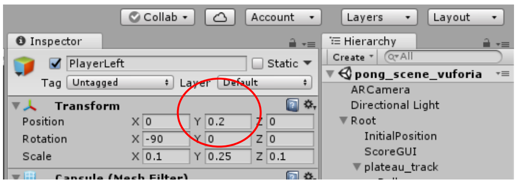
Constraints

Freeze Position ☑ X ☑ Y ☑ Z

Freeze Rotation ☑ X ☑ Y ☑ Z

  - Beware of relative object placements. For example, objects attached to markers on the ground should be placed slightly above, especially if another marker is linked to a playground with some thickness:





☑ PlayerLeft □ Static

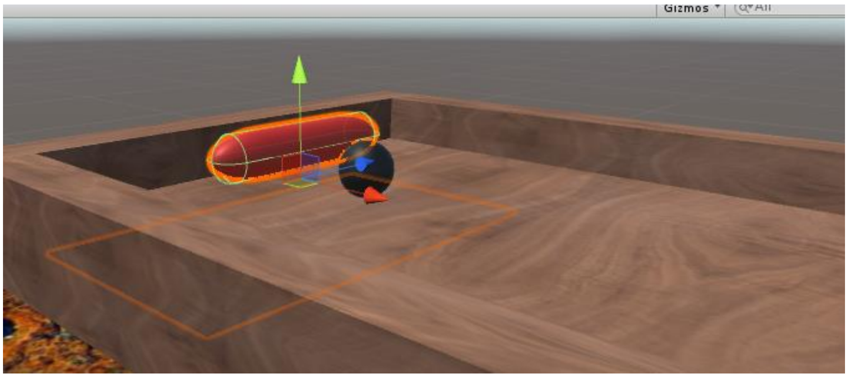
Tag Untagged Layer Default

Transform

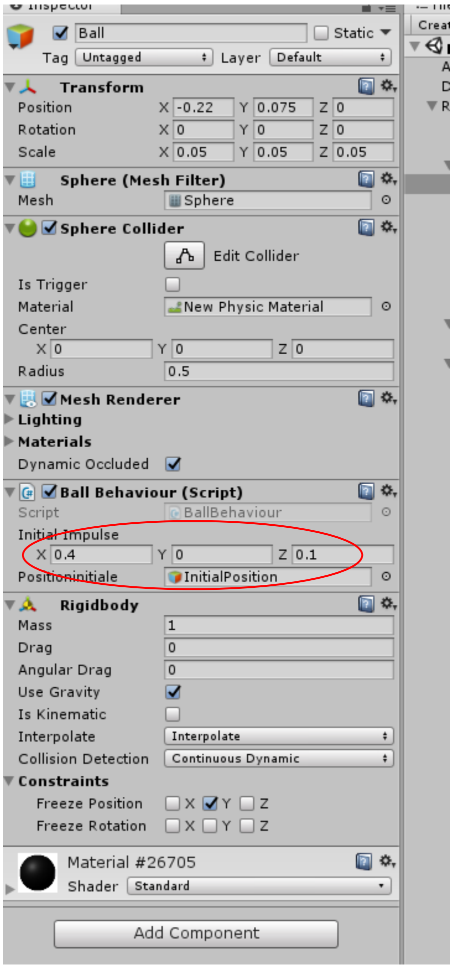
Position	X 0	Y 0.2	Z 0
Rotation	X -90	Y 0	Z 0
Scale	X 0.1	Y 0.25	Z 0.1

Capsule (Mesh Filter)

pong_scene_vuforia
ARCamera
Directional Light
Root
InitialPosition
ScoreGUI
plateau_track



- ○ Any moving object (with proper scale!) should be fast enough to produce good collisions, but not too fast:



☑ Ball □ Static

Tag Untagged Layer Default

Transform

Position	X -0.22	Y 0.075	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 0.05	Y 0.05	Z 0.05

Sphere (Mesh Filter)

Mesh: Sphere

☑ Sphere Collider

Edit Collider

Is Trigger □

Material: New Physic Material

Center: X 0 Y 0 Z 0

Radius: 0.5

☑ Mesh Renderer

Lighting

Materials

Dynamic Occluded ☑

☑ Ball Behaviour (Script)

Script: BallBehaviour

Initial Impulse: X 0.4 Y 0 Z 0.1

Positioninitiale: InitialPosition

Rigidbody

Mass	1
Drag	0
Angular Drag	0
Use Gravity	☑
Is Kinematic	□
Interpolate	Interpolate
Collision Detection	Continuous Dynamic

Constraints

Freeze Position □ X ☑ Y □ Z

Freeze Rotation □ X □ Y □ Z

Material #26705

Shader: Standard

In conclusion, making physics work in AR can be a bit tricky. In Unity, things were setup to allow games to work correctly at “human” scales (dimensions in the meter to kilometer range), with relatively high speeds (1m/s is already fast for a small object). However, marker-based AR is not yet easily done (practically) with these dimensions. Things need to be scaled-down a bit, but this is not default behavior. Caution advised.