Fonaments de modelatge i animació

AA3

Roger Aguilar Emanuel
Èric Ruiz Cuscó
3er CDI

Exercice location

(object, script, function)

Exercici 1:
1- IK_tentacles, NotifyShoot()
2- BlueTarget, MovingTarget, Update()
3- IK_Scorpion, Update()
4- Main Camera, Reset, Update()

Exercici 2:
1- IK_Scorpion, Update()
2- MovingBall, OnCollisionEnter()
3- MovingBall, ComputeTrajectory() & Update()
4- MovingBall, Update()

Exercici 3:
1-IK_Scorpion, UpdateFutureLegBases()
2-inside the obstacles object in the hierarchy
3-MyScorpionController(DLL) LerpLegs()
4-IK_Scorpion UpdateBodyPosition()
5-IK_Scorpion UpdateBodyRotation()
6-ScorpioMovement Update()

Exercici 4:
1- MyScorpionController, updateTail() & ApproachTarget(Vector3 target)
2- MyScorpionController, ApproachTarget(Vector3 target) & NewErrorFunction()

Exercici 5:
1- MyOctopusController, update_ccd()
2- IK_tentacles, NotifyShoot()
BlueTargetFollower, TargetFollower, StartFollowingTarget()

Explanations

Exercici 4:
1- For the tail animation we made it adjust to the magnus effect parameter by adding an offset to the target's X axis, hitting the ball by the right or left side when needed. When it comes to the strength of the shot, we had to make the animation go faster, so to do that we multiplied the result of the gradient calculation by the slider value.

3- We tried to implement a new error function to the gradient descent but we were not able to make it work properly.

Exercici 2:
- Angular Velocity:
impactVect = (contactPoint - transform.position).normalized
angularMomentum = CrossProduct(impactVect, velocity.normalized)
angularVel = angularMomentum * (effectStrength.value * -1)

The effectStrength is the slider we use for the strength and the contactPoint the contact between the tail and the ball.

With the dot between the impactVect and the velocity we get the angular momentum.

- Magnus Force:

$$L = p * vfree * (2\pi r)^2 * f * l$$

L – Magnus force in N
P- density of surrounding fluid kg/m^3
vFree - the free stream velocity of the fluid in m/s
R-the radius
L- length
f-angular velocity

L = airDensity * freeStream * Mathf.Pow((2.0f * Mathf.PI * 0.25f), 2) * 0.5f * angularVel

We chose this formula because we think is in the perfect balance between being complex enough and not too demanding computationally.