ACE Replay

User Guide

State base replay system developed for Unity

by

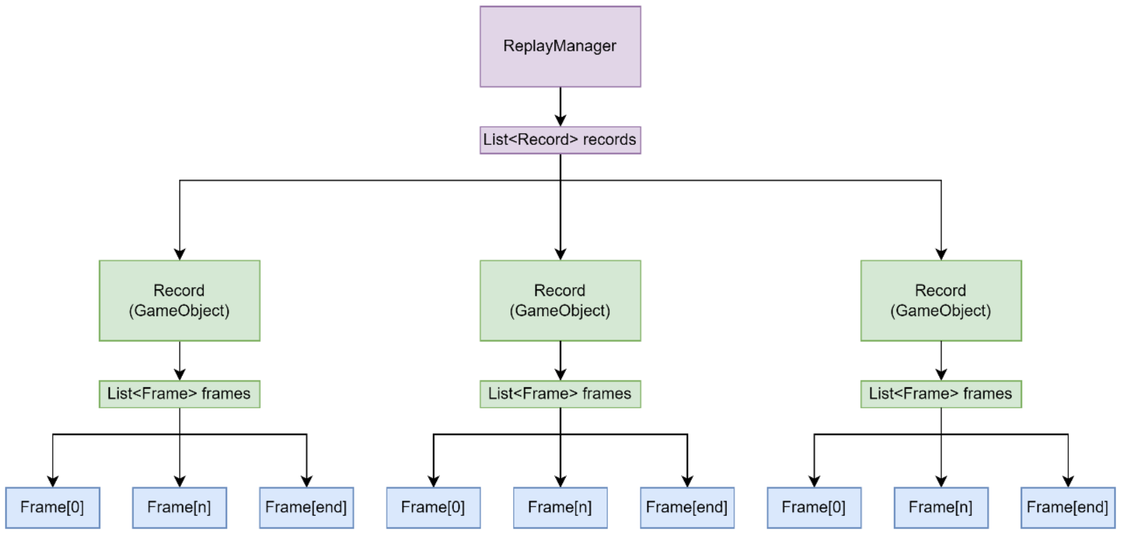
Josep Lleal

Features:

- Scripts and prefabs to integrate the system quickly and easily.
- Specify easily what should be recorded by dragging and dropping the script to the desired object, without the need to program.
- Example scenes to see how the system works and its utilities.
- Recording of transforms, animations, particles, and audios.
- Supports instantiation and deletion of recorded objects.
- Optimization to record at low frame rates without losing smoothness of replay, thanks to interpolation.
- Simple UI to control instant replay.
- Replay can be seen from different camera angles using the fly-around replay camera. The replay can also be seen from any of the existing scene cameras, including the gameplay camera.
- Supports playback at different speeds: x0.25, x0.5, x1, x2, x4.
- Supports frame-by-frame playback and reverse playback.
- Travel back in time mechanic.
- Fully C# commented code that can be extended upon need.

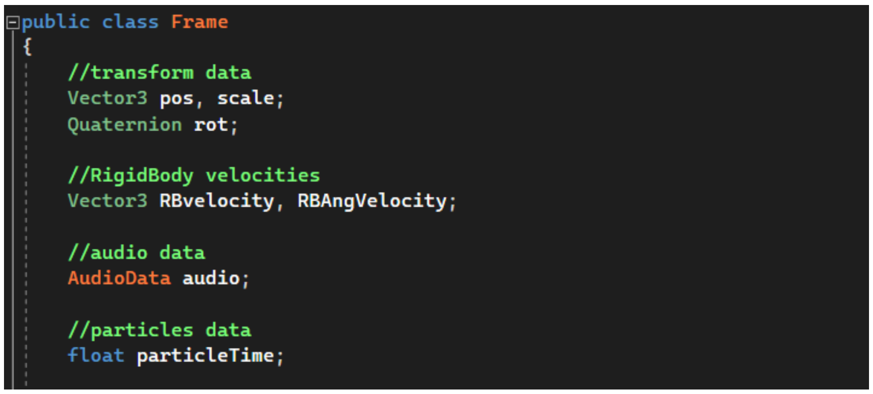
Main System:

Ace Replay system is made up of three different classes (ReplayManager, Record, and Frame), which work all together to successfully replicate the gameplay that was previously recorded. This system can be integrated into any Unity project by only dragging and dropping the prefab of the ReplayManager to the scene, as well as adding the record class to all objects that have to be recorded.



Frame:

Frame, is a data structure used to hold information from a recorded frame. This data contains information of the transforms, audio data, particles, and rigidBodies velocities.

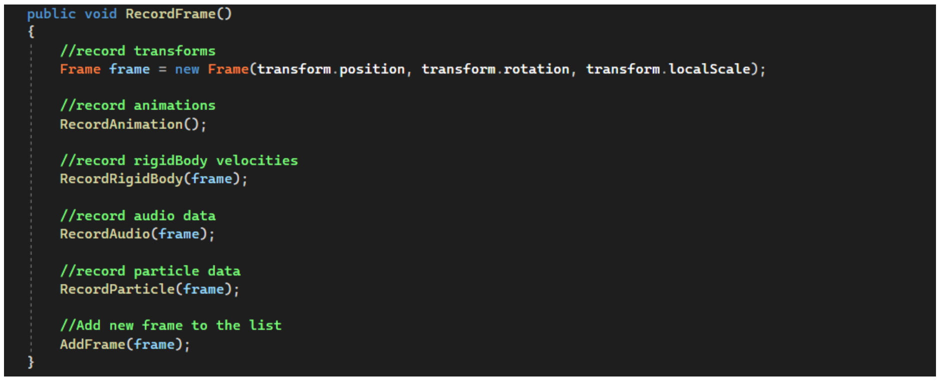


```
public class Frame
{
    //transform data
    Vector3 pos, scale;
    Quaternion rot;

    //RigidBody velocities
    Vector3 RBvelocity, RBAngVelocity;

    //audio data
    AudioData audio;

    //particles data
    float particleTime;
```

Record:

The record class is the component that is **attached to an object to determine that it has to be recorded**. The ReplayManager of the scene has to be assigned to this component by either drag and drop or by writing the ReplayManager object's name.

One of the main functions of this class is the RecordFrame(), which creates a Frame with all its information at that instance, and is stored in Record's list. The call of this function is done in the ReplayManager. By default, if the game is running, and it is not in replay mode, the gameplay will be recorded.

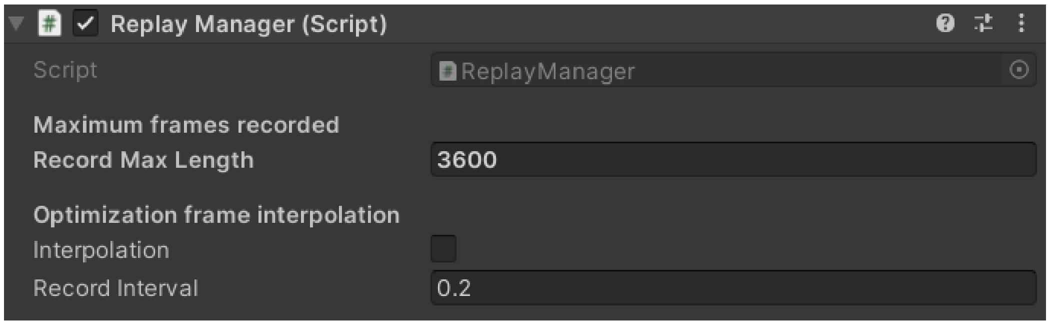


```
public void RecordFrame()
{
    //record transforms
    Frame frame = new Frame(transform.position, transform.rotation, transform.localScale);

    //record animations
    RecordAnimation();

    //record rigidBody velocities
    RecordRigidBody(frame);

    //record audio data
    RecordAudio(frame);

    //record particle data
    RecordParticle(frame);

    //Add new frame to the list
    AddFrame(frame);
}
```

ReplayManager:

The ReplayManager is the main component of the system and it is **responsible for recreating the recorded gameplay**. This can be done because the ReplayManager has a list with all the Records of the recorded objects.

When implementing the system to a Unity project, **a ReplayManager must be placed in the scene**. To do so, you can use the created Prefab called ReplayManager. Once it is in the scene, some settings can be changed, like the maximum length of the record, if interpolation optimization is wanted, as well as the interval in seconds between recorded frames.



Replay Manager (Script)

Script: ReplayManager

Maximum frames recorded

Record Max Length: 3600

Optimization frame interpolation

Interpolation: ☐

Record Interval: 0.2

These are the main functions that can be called from the **ReplayManager** to manage the **Instant Replay:**

- **ReplayMode():** Returns a boolean if the manager is in replay mode or not.
- **EnterReplay():** Enters replay mode.
- **QuitReplayMode():** Exits replay mode.
- **RestartReplay():** Start the replay from the beginning.
- **PauseResume():** To stop and resume the replay.
- **GoForward()** and **GoBack():** To advance and go back one frame.
- **SpeedUp()** and **SpeedDown():** Changes the replay speed playback.
- **NextCamera()** and **previousCamera():** To change the active camera

As a demonstration utility of the system, a **travel back mechanic** has been implemented as well. These are the functions inside **ReplayManager** to manage the travel back in time mechanic:

- **StartTravelBack(float t):** Enters replay mode and plays a travel back of *t* seconds.
- **ExitTravelBack():** Exits replay mode and the game is resumed at the exited frame.
- **StartTravelBack():** Starts a travel back until ExitTravelBack() is called or until the end of the record is reached.

Finally, to **keep track of deleted objects**, the **ReplayManager** has the next function:

- **DestroyRecordedGO(GameObject obj):** This function fakes the deletion of a recorded object by disabling it. Once the replay is exited, the object will be destroyed for real.

Replay Aspects:

When adding the ACE Replay system into your game, some things should be taken into consideration:

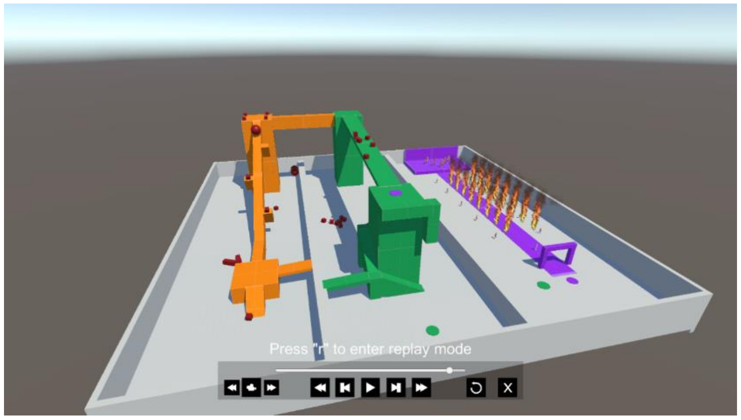
- **ACE Replay system works for 3D projects,** this asset tool was developed specifically for 3D games. If you still want to give it a try, feel free to make the necessary changes to make it work for 2D games.
- **On exiting a replay, all the recorded frames will be cleared.** This is one of the most important aspects to take into consideration, as it is quite limiting for your game. At the moment of the release, ACE Replay is using Unity's animator recorder to handle animations. There are two problems with this recorder: It has a limit of 10000 recorded frames and, when the recording of animations starts, all the previous recorded frames are deleted.
- **Scripts of recorded objects are disabled during replay.** To ensure a good replication of objects during replay, it is important to make sure that no scripts get in the way of the replay. For that reason, when replay mode is entered, all the recorded object scripts will be disabled. Keep in mind that, if you don't want to disable a certain script, in the record component there is a serialized list where you can place that script and it will not be disabled.
- **RigidBodies are disabled during replay.** Similar to the scripts, when in replay mode we don't want external systems to interfere with the replication. This is why, during the replay mode all rigidBodies will be set to isKinematic = true, and when exited, they will be set back to false.
- **If a recording object has to be deleted,** the method from the ReplayManager DestroyRecordedGO() must be called instead of the default Unity's Destroy().

Replay Example Scenes:

ACE Replay system package comes along with two example scenes to showcase the utilities of the system. There is the instant replay example scene and the travel back in time example scene.

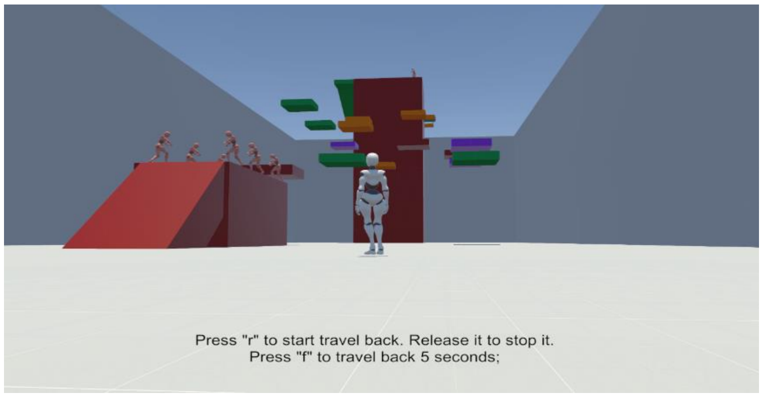
Instant Replay Example Scene:

This scene was made to show the instant replay functionality. In this scene, you can control a player to move around the map, which is a circuit with different elements. Through the map, different objects will be spawned and deleted, some audios will be played alongside some particles and animations, all to prove that the ACE Replay system can handle all of it.



Travel Back in time Example Scene:

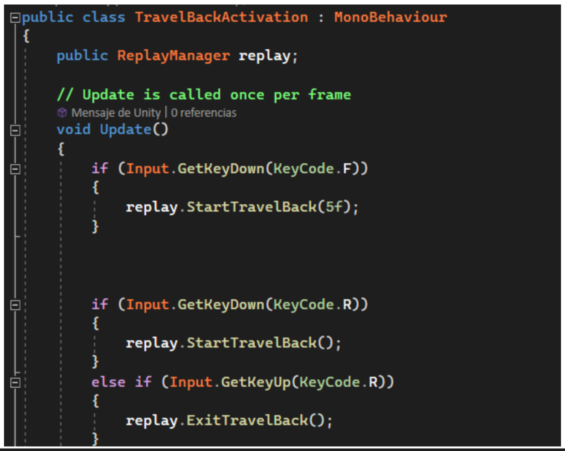
This second scene was made to show the travel back in time mechanic that can be done with this replay system. The scene features a parkour course where you have to reach the top of the big pilar. If at any point of the course you fall, you can go back in time to try again the failed jump.


Press "r" to start travel back. Release it to stop it.
Press "f" to travel back 5 seconds;

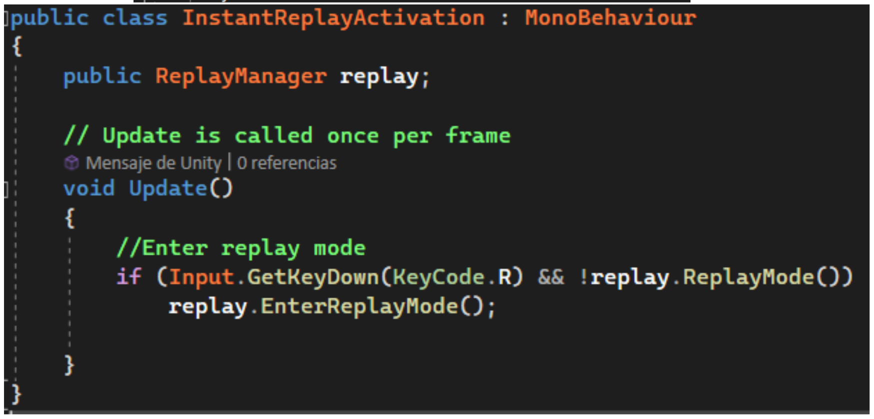
Integration step by step:

One of the purposes of this project was to make the replay system implementation as simple as possible. After its development, I can say that ACE Replay can be set up very quickly with only three steps:

- **STEP ONE:** Import the ReplayManager prefab into your scene. It is highly recommended to do it this way because the ReplayManager prefab contains as children, all of the UI of the instant replay.
- **STEP TWO:** Identify all the objects that have to be recorded and add a Record component to them. When doing this, assign the ReplayManager by either making sure the ReplayManager name is the same as the prefab (which should be by default) or drag and drop the ReplayManager from the scene hierarchy.
- **STEP THREE:** Finally, the only thing left to do is call the necessary functions to activate the instant replay or travel back in time. The next images are scripts already created in the packet to activate both of the utilities in their respective levels.



```
public class TravelBackActivation : MonoBehaviour
{
    public ReplayManager replay;

    // Update is called once per frame
    Mensaje de Unity | 0 referencias
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.F))
        {
            replay.StartTravelBack(5f);
        }


        if (Input.GetKeyDown(KeyCode.R))
        {
            replay.StartTravelBack();
        }
        else if (Input.GetKeyUp(KeyCode.R))
        {
            replay.ExitTravelBack();
        }
```

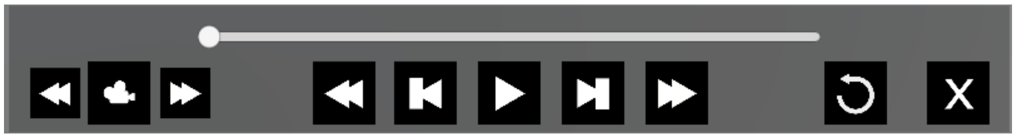


```
public class InstantReplayActivation : MonoBehaviour
{
    public ReplayManager replay;

    // Update is called once per frame
    Mensaje de Unity | 0 referencias
    void Update()
    {
        //Enter replay mode
        if (Input.GetKeyDown(KeyCode.R) && !replay.ReplayMode())
            replay.EnterReplayMode();

    }
}
```

Instant Replay Controls:

When using the instant replay, these are the functionalities and controls available.

Instant Replay UI buttons:



Using the image from above as a reference, from left to right these are the functions of the UI buttons:

- **Use the previous camera and use the next camera:** With these two buttons the user can use any camera that is in the recorded scene, including the replay camera which can be moved freely.
- **Speed down:** To slow down the replay's playing speed. There are two slow-motion speeds, x0.5 and x0.25.
- **Go back one frame:** This button lets the user go back exactly one frame and the replay is paused.
- **Pause and resume:** When this button is pressed, if the replay is paused it will start playing. It also works the other way around, changing from playing to paused.
- **Advance one frame:** This button lets the user go forward exactly one frame and the replay is also paused.
- **Speed up:** To increase the playing speed of the replay. The replay speed can be increased to x2 and x4.
- **Restart replay**: When this button is used, the replay will start again from frame 0 ergo the beginning.
- **Exit replay:** this button is used to exit the replay mode.
- **Slider**: Slider to move to a specific time of the replay.

Fly around camera controls:

- **A** - Move left.
- **D** – Move right.
- **W** – Move forward.
- **S** – Move backward.
- **E** – Move up.
- **Q** – Move down.
- **Mouse right-click** – Rotate camera.

Player controls:

- **A** - Move player left.
- **D** - Move player right.
- **W** - Move player forward.
- **S** - Move player backward.
- **Space Bar** – Jump.
- **Mouse movement** - Rotate player camera.

Replay system record features:

For the first release, ACE Replay system only supports the record of transforms, animations, audios, and particles, as well as the handling of instantiated and deleted objects. Keep in mind that this tool was done by a single person and was my final bachelor’s degree project, so I was time limited.

If for your game a new feature needs to be added, feel free to extend the code to your needs.

Request features:

If you are not good at programming or you struggle to implement a new feature, feel free to request a new feature. If there is a great demand to add new features, I will try to implement them for next possible releases.

Report bugs:

If while using the ACE Replay system you encounter a bug and you want it to get fixed, feel free to report it. When reporting a bug make sure to describe properly what is the problem and how can it be reproduced.

Contact to request features, report bugs or get some help:

- **Mail:** acereplaysystem@gmail.com
- **Twitter:** https://twitter.com/ace_replay

Contact the author:

For business inquiries you can contact me through:

- **LinkedIn**: https://www.linkedin.com/in/josep-lleal-sirvent/
- **Twitter:** https://twitter.com/xPepe_12