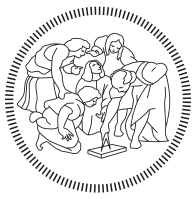
Anno Accademico 2020/2021



Report Progetto di Ingegneria Informatica

Editor di attuazioni temporizzate per robot animatronici

Davide Veronesi

Referente

Andrea Bonarini

Titolare

Gianluca Palermo

09/09/2021

Indice

1 Specifica

La specifica richiede l'implementazione di un programma per gestire attuazioni temporizzate per robot realizzati con schede Arduino/Raspberry Pi.

Il programma dovrà supportare diversi tipi di attuatori con relative impostazioni.

Dovrà essere presente la possibilità di salvare e ricaricare le diverse animazioni create.

2 Implementazione

È possibile creare due tipi di oggetti:

- Divisori;
- Attuatori.

I **divisori** sono degli oggetti che migliorano l'organizzazione del lavoro in quanto permettono di raggruppare più attuatori insieme, in modo da riprodurre contemporaneamente le sequenze di animazioni previste da questi ultimi. La riproduzione si interrompe quando tutte le sequenze previste dagli attuatori giungono al termine, perciò i divisori risultano essere degli strumenti utili, soprattuto in fase di testing.

Gli **attuatori** sono i blocchi fondamentali che permettono di predisporre l'intervallo massimo di valori a cui i *keyframes* possono essere impostati. Infatti, all'interno di un attuatore, l'utente ha l'abilità di aggiungere *keyframes*, ovvero sottoblocchi che specificano l'animazione desiderata.

I tipi disponibili di Attuatore sono:

- **Digital**: il più semplice con solo due stati (acceso/spento);
- **Analog**: può avere un range di valori;
- **Servo**: per controllare **servomotori**;
- **Buzzer**: per controllare **buzzer** attivi o passivi;
- **RGB**: per controllare facilmente i **LED RGB** al posto di usare tre diversi attuatori di tipo Analog;

Le aree evidenziate hanno diverse funzioni:

- Area arancione - **Option Area**: contiene il nome, i pin, il tipo di attuatore, il valore massimo e il frame rate;
- Area azzurra - **Test Area**: contiene il necessario per inviare un singolo valore alla scheda, quindi testarlo;
- Area fucsia - **Keyframes Area**: contiene i keyframes associati all'attuatore e permette la loro modifica;

Premendo il tasto `Play`, il programma genera i frames mancanti per la produzione della transizione richiesta tra un keyframe e l'altro, successivamente tutti valori, sia espressi dall'utente sia computati dal programma, vengono inviati alla scheda mediante comunicazione seriale in formato JSON.

Il programma sulla scheda Arduino si limita ad interpretare i dati ricevuti, così che un successivo inserimento di nuovi attuatori risulti molto semplice, poiché richiederà solo l'implementazione di un nuovo *case* e della rispettiva funzione per l'attuatore, senza dover modificare nessun altro pezzo di codice.

Il salvataggio (*save*) avviene su un file di tipo JSON dove vengono memorizzate tutte le informazioni necessarie per ricreare gli stessi attuatori e al contempo salva su un altro file tutti i keyframe con i relativi valori.

Il caricamento (*load*) non sovrascrive gli attuatori già creati, ma ne aggiunge di nuovi in fondo alla lista, ciò permette di duplicare un determinato attuatore.

3 Tecnologie usate

3.1 Unity

Il programma è stato realizzato all'interno dell'ambiente di sviluppo **Unity**. Questa scelta è stata presa poiché l'implementazione di un'interfaccia grafica utilizzando gli strumenti base di Unity risulta molto agevole, grazie anche all'esperienza pregressa con tale software, ed è possibile generare eseguibili per varie piattaforme, tra cui Windows/Mac/Linux/Android/iOS.

Il caricamento e il salvataggio sono realizzati utilizzando file **JSON**, così come la comunicazione tra il programma e la scheda.

3.2 Arduino

La libreria usata per deserializzare JSON è **ArduinoJson**, la quale permette di inviare/ricevere file JSON sia tramite comunicazione seriale sia tramite ethernet e Wi-Fi.

Poichè l'animazione è salvata come JSON, essa occupa molta memoria pertanto risulta impossibile inserirne di complesse su schede il cui spazio di archiviazione è fortemente limitato.

Una soluzione ottimale risulta essere il collegamento della scheda Arduino con un **Raspberry Pi** (o equivalente), in cui l'Arduino riceverà solo le informazioni e le eseguirà senza svolgere alcun tipo di computazione intensiva che potrebbe causare problemi con la sincronizzazione degli attuatori.

4 Installazione

4.1 Requisiti

- **Unity 2020.1.1** o superiore;
- **ArduinoJson 6** o superiore;

4.2 Procedura

4.2.1 Unity

Per poter aprire il progetto in Unity è necessario scaricare o clonare la repo; usando Unity Hub si può comodamente identificare il progetto per aggiungerlo alla propria lista.

Se non c'è necessità di modificare il programma, basterà scaricare dalla cartella Release la cartella con l'eseguibile per il proprio sistema operativo e avviare quest'ultimo.

4.2.2 Arduino

Per quanto riguarda la scheda Arduino, è necessario caricarvici sopra il codice fornito e successivamente collegarlo al programma selezionando la porta corretta. Se non dovesse essere possibile collegarla, staccare e riattacare la scheda in modo che il programma abbia la priorità sulla porta selezionata.

5 Link utili

- **Github repository: https://github.com/Verdax97/Syncronino**
- **Arduino: https://www.arduino.cc**
- **ArduinoJson: https://arduinojson.org**
- **Unity: https://unity.com**

6 Appendice

6.1 Glossario

- **Attuatore**: dispositivo che consente di intervenire indirettamente sul funzionamento o sul controllo di meccanismi.
- **Servomotori**: è un particolare tipo di motore, generalmente di piccola potenza, che si differenzia dai motori tradizionali in quanto le sue condizioni operative sono soggette ad ampie e spesso repentine variazioni sia nel campo della velocità sia nel campo della coppia motrice.
- **Buzzer**: è un dispositivo di segnalazione audio.
- **LED**: Light Emission Diode (Diodo ad Emissione di Luce).

6.2 Esempi di Json

Esempio di Keyframe passato dal programma all'Arduino:

```
{
    "active": true,
    "timing": 0.0,
    "duration": 0.0,
    "fade": 1,
    "values": [
        0
    ],
    "type": 97,
    "pins": [
        9
    ]
}
```

Esempio di Animazione salvata di un attuatore digitale:

```
{
    "actuators": [
        {
            "name": "d",
            "typeComponent": "Actuator",
            "typeActuator": "d",
            "pins": [
                8
            ],
            "maxValue": 1,
            "rate": 1,
            "keyframes": [
                {
                    "active": true,
                    "timing": 0.0,
                    "duration": 0.0,
                    "fade": 0,
                    "values": [
                        1
                    ]
                }
            ]
        }
    ]
}
```