Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo"
Consiglio Nazionale delle Ricerche

ISTI Technical Reports

Una metodologia di sviluppo di applicazioni di realtà aumentata per i beni culturali applicata ad un caso di studio: il Parco di Pinocchio

Fabrizio Matarese, ISTI-CNR, Pisa, Italy
James Magnavacca, Università di Pisa, Pisa, Italy
Massimo Magrini, ISTI-CNR, Pisa, Italy

ISTI-TR-2021/004

Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo"
Consiglio Nazionale delle Ricerche

Una metodologia di sviluppo di applicazioni di realtà aumentata per i beni culturali applicata ad un caso di studio: il Parco di Pinocchio

Matarese F., Magnavacca J., Magrini M.

ISTI-TR-2021/004

Il documento descrive la metodologia di sviluppo delle applicazioni di realtà aumentata adottata nel progetto VERO, Virtualità Interattiva nel Parco di Pinocchio.

Keywords: Augmented reality, Cultural heritage.

Citation

Matarese F., Magnavacca J., Magrini M., *Una metodologia di sviluppo di applicazioni di realtà aumentata per i beni culturali applicata ad un caso di studio: il Parco di Pinocchio*. ISTI Technical Reports 2021/004. DOI: 10.32079/ISTI-TR-2021/004.

Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo"
Area della Ricerca CNR di Pisa
Via G. Moruzzi 1
56124 Pisa Italy
http://www.isti.cnr.it

Istituto di Scienza e Tecnologie dell'Informazione

Consiglio Nazionale delle Ricerche

Fabrizio Matarese

James Magnavacca

Massimo Magrini

Una metodologia di sviluppo di applicazioni di Realtà Aumentata per i beni culturali applicata ad un caso di studio: il Parco di Pinocchio.

Sommario

1 Introduzione

Le applicazioni informatiche applicate al patrimonio culturale si concentrano principalmente sul miglioramento dei processi di digitalizzazione e conservazione digitale. In passato le moderne tecnologie erano utilizzate solo da professionisti, come archeologi, architetti ecc.,recentemente invece, sempre più musei, siti archeologici e mostre hanno iniziato a esplorare l'uso delle nuove tecnologie per creare nuovi tipi di interazione con i beni culturali e migliorare l'esperienza dell'utente.

Diverse soluzioni interattive sono utilizzate nell'ambito del Patrimonio Culturale, come ad esempio sistemi di realtà virtuale/aumentata (VR/AR), mondi virtuali 3D, ecc. L'adozione della realtà aumentata nel patrimonio culturale è iniziata già nel 1999 con il sistema di realtà aumentata mobile del progetto MARS (Mobile augmented reality systems)[1]. Il prototipo di questo progetto utilizza diverse interfacce utente per consentire agli utenti di accedere e gestire le informazioni registrate spazialmente sulle coordinate del mondo reale. Gli utenti possono sperimentare presentazioni multimediali spazializzate che vengono visualizzate su un visore trasparente indossato sulla testa, utilizzato insieme a un computer portatile con penna digitale.

[1]Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., Hallaway, D.: Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system. Comput. Graph. **23**(6), 779–785 (1999).

Figura 1: Prototipo MARS indossato da un utente.

Figura 2: Un'immagine catturata dal visore trasparente indossato sulla testa con informazioni in Realtà Aumentata (progetto MARS).

ARCHEOGUIDE (Augmented Reality-Based Cultural Heritage On-Site Guide) è un altro progetto del 2002, che utilizzava primitive tecnologie AR in un prototipo presso il sito archeologico greco di Olimpia[2].

[2]Vlahakis, V., Ioannidis, N., John, K., Tsotros, M., Gounaris, M.: Archeoguide: an augmented reality guide for archaeologicalsites. Comput. Graph. Art Hist. Archaeol. **22**(5), 52–60 (2002).

L'hardware era basato su un PC portatile Toshiba di fascia alta e un display montato sulla testa (HMD) di Sony Glasstron con trasparenza variabile su cui appaiono i mondi AR. L'utente indossa un casco da bicicletta con una webcam USB, una bussola digitale montata sulla parte superiore e uno zaino contenente il laptop, il ricevitore DGPS, la batteria, il modulo di distribuzione dell'alimentazione e l'hardware WLAN.

Figura 3: Sistema AR utilizzato per il progetto ARCHEOGUIDE.

Figura 4: Esempio di rendering: il tempio di Hera ricostruito digitalmente (ARCHEOGUIDE).

In meno di 20 anni, l'ambito delle tecnologie mobile ha attraversato un profondo e rapido sviluppo. Quello che 20 anni fa era realizzabile solo con un hardware molto

pesante da trasportare (e con risultati estremamente imprecisi) sta diventando possibile tramite dispositivi molto più comodi e leggeri.

Le tecnologie AR/VR sono attualmente utilizzate sempre di più nel patrimonio culturale e con approcci spesso diversi. Di seguito forniamo alcuni esempi di queste applicazioni.

Il progetto *Terra Mosana*[3] è una collaborazione tra comuni, siti del patrimonio, musei e università per rafforzare l'attrattiva turistica dell'Euroregione Mosa-Reno (EMR), un'area con una ricca storia divisa tra Belgio, Germania e Paesi Bassi. Questo processo di valorizzazione sarà effettuato attraverso narrazioni digitali della storia condivisa di diverse città nell'area. Si baserà sullo sviluppo di modelli 3D del patrimonio archeologico e urbano fruibili in realtà virtuale e in realtà aumentata. In ogni città partner verranno proposte al pubblico esperienze in realtà aumentata e virtuale (utilizzando dispositivi interattivi quali smartphone, tablet, visori per realtà virtuale, ecc.) per riscoprire tratti di storia comune all'interno dei siti del patrimonio. Questo progetto è particolarmente interessante perché presenta una rete di piccoli siti, collegati tra loro in percorsi tematici grazie alle tecnologie AR/VR.

Le tecnologie virtuali sono state spesso utilizzate per proporre siti antichi che non esistono più, utilizzandone piccole parti per attivare la ricostruzione digitale di interi scenari. *Hidden Florence 3D*[4] è un progetto realizzato in collaborazione tra la National Gallery, Calvium, l'Università di Exeter, l'Università di Cambridge e lo studio Zubr di Bristol. Con quest'app si può esplorare Firenze alternando tra una mappa moderna e una superbamente dettagliata del XVI secolo: una riproduzione digitale del cartografo Stefano Bonsignori, *Nova pulcherrimaecivitatisFlorentiaetopographia accuratissima delineata*.

Inoltre, questo progetto presenta speciali contenuti in AR,relativi al patrimonio culturale in modo da ripristinare virtualmente una chiesa fiorentina distrutta da tempo, la chiesa di San Pier Maggiore demolita nel 1784. Il modello 3D dà vita alla chiesa con dettagli, texture e illuminazione accurati. *Hidden Florence 3D* propone un'esperienza di realtà aumentata per ricostruire virtualmente il contesto originario di un'opera d'arte: la pala

[3]https://www.terramosana.org/
[4]https://hiddenflorence.org/

d'altare realizzata da Jacopo di Cione e conservata alla National Gallery di Londra. Il risultato è un'esperienza potente che consente a chiunque di "camminare" all'interno della chiesa come facevano gli abitanti di Firenze nei secoli scorsi. Utilizzando l'app su iPhone o iPad e avvicinandosi alla pala d'altare si attiverà una transizione mozzafiato, immergendo il visitatore in un ricostruzione 3D degli interni della chiesa.

Le tecnologie di realtà aumentata possono aiutare a spiegare come funzionano oggetti con un elevato grado di complicazione. AVRLab (Università del Salento) ha presentato un'applicazione di realtà aumentata per facilitare la comprensione del funzionamento delle macchine di Leonardo attraverso una forma di visualizzazione immediata che può essere utilizzata con uno smartphone o una webcam. L'app AR, *Scienza Machinale Aumentata*[5], è stata sviluppata per la mostra *Il genio tra le pagine: le macchine di Leonardo in biblioteca* nel 2016 come supporto multimediale per rendere più chiara e immediata la comprensione delle macchine di Leonardo.

La stessa Università ha presentato un'applicazione basata sulla tecnologia di Realtà Aumentata per valorizzare alcune aree archeologiche pugliesi, riguardanti la popolazione autoctona della regione Puglia. L'obiettivo è quello di migliorare la conoscenza dei visitatori su questa popolazione, di comprendere meglio le aree archeologiche del Museo Diffuso "Castello di Alceste" a San Vito dei Normanni (BR), e la sede del Fondo Giuliano, a Vaste (LE)[6]. Sulla base di una mappa planimetrica delle aree archeologiche, i modelli 3D vengono sovrapposti e combinati con altri tipi di contenuti (immagini e testi) per consentire ai visitatori di comprendere maggiormente la storia e le caratteristiche dei siti antichi. Questo progetto è un buon esempio del fatto che le tecnologie di realtà aumentata possono essere utilizzate anche insieme a materiali culturali forniti in coordinamento durante le visite a siti culturali, come stampe o opuscoli.

Un'altra funzionalità della Realtà Aumentataconsiste nell'aggiungeretexture virtuali su oggetti organici, come scheletri di animali. Ad esempio, a Washington, il Museo

[5]http://avrlab.it/scienza-machinale-aumentata/
[6]http://avrlab.it/augmented-reality-for-the-enhancement-of-apulian-archaeological-areas/

Nazionale di Storia Naturale ha proposto l'app *Skin and Bones AR*[7]. Utilizzando quest'app, puoi sovrapporre la pelle allo scheletro di vari animali, migliorando notevolmente l'esperienza del visitatore.

Un altro esempio interessante è il *Progetto VERO* (ISTI-CNR e Fondazione Collodi). Il "Parco di Pinocchio", nato dal celebre libro *Le avventure di Pinocchio* di Carlo Lorenzini, permette ai suoi visitatori di vivere le emozioni e gli eventi del libro italiano più amato e conosciuto nel mondo. All'interno del "Parco di Pinocchio" (situato a Collodi nel comune di Pescia), la *Piazzetta dei Mosaici* contiene pregevoli fregi mosaicati realizzati dallo scultore toscano Venturino Venturi. Gli operatori del progetto intendono utilizzare un sistema tecnologico non invasivo che ne esalti il valore. La soluzione proposta prevede l'utilizzo di tecniche di realtà aumentata per rendere più coinvolgente la visita alla Piazzetta senza stravolgerne l'aspetto con elementi estranei. Inquadrando le scene dei mosaici con un paio di occhiali a realtà aumentata o con un comune smartphone, il visitatore potrà visualizzare un'animazione tridimensionale (creata appositamente da artisti digitali) integrata con la scena reale, offrendo l'illusione che il mosaico prenda vita nello spazio. Le animazioni sono inserite in una sorta di gioco a quiz, in cui i visitatori devono scegliere i vari mosaici in una sequenza corretta per completare il gioco. Pur mantenendo intatto il suo valore storico-artistico, il sito si trasforma in un gioco educativo interattivo grazie alla tecnologia.

[7]https://www.dianamarques.com/portfolio/augmented-reality-app-skin-bones/

2 Descrizione degli strumenti di sviluppo

2.1 Unity 3D

Unity 3D è un motore grafico multipiattaforma sviluppato da Unity Technologies che permette lo sviluppo di molteplici applicazioni: videogiochi, film e animazioni 3D, modelli architettonici, simulazioni in realtà virtuale (VR) e realtà aumentata (AR), soluzioni per l'industria automobilistica e progetti educativi basati sui nuovi media.

L'ambiente di sviluppo di Unity è compatibile con Windows, con MacOs e con Linux ma è possibile esportare il progetto per un gran numero di piattaforme: Microsoft Windows, macOS, Linux, Xbox 360, PlayStation 3, PlayStation Vita, Wii, iPad, iPhone, Android, Windows Mobile, PlayStation 4, Xbox One, Wii U e Switch. Si può inoltre produrre un'applicazione web capace di girare su ogni browser che supporti il plug-in Unity Web Player.

Possiamo osservare subito che uno dei punti di forza di questo motore grafico sta nella sua versatilità sia per quanto riguarda le tipologie di progetto che si intende sviluppare sia per le piattaforme compatibili: lo stesso progetto infatti può essere esportato su molteplici piattaforme.

Il linguaggio di programmazione che questo motore utilizza per applicare la logica ai progetti è C#, anche se è attualmente in preview un sistema chiamato DOTS (Data-Oriented Technology Stack) che, è meno intuitivo rispetto a C# ma permette di sfruttare la potenza dei nuovi processori multicore e di ottimizzare le performance.[8]

[8]https://unity.com/dots

2.1.1 Interfaccia dell'Editor di Unity

L'interfaccia dell'Editor di Unity è composta di varie finestre, ognuna con le sue funzioni, più una barra degli strumenti.

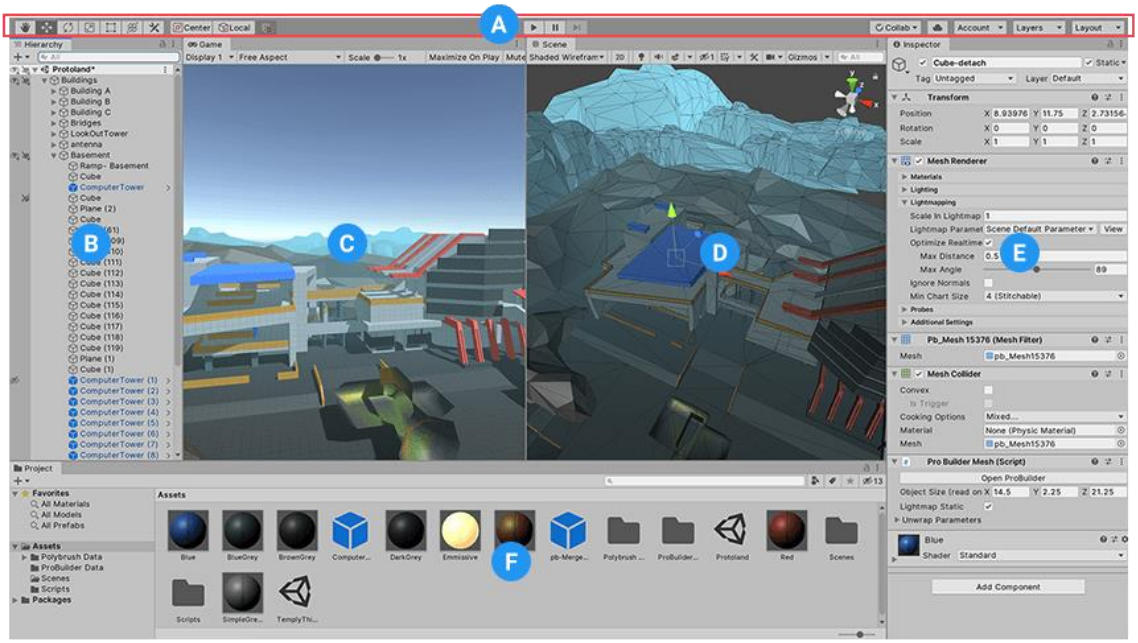


Figura 5: Interfaccia dell'Editor di Unity.

In fig.5, tratta direttamente dal manuale di Unity[9], possiamo osservare le varie finestre che compongono l'interfaccia dell'Editor.

(A) La barra degli strumenti fornisce l'accesso alle funzioni più essenziali. Centralmente abbiamo tre bottoni chepermettono di premere play, per testare il gioco o il progetto a cui stiamo lavorando, pausa e stop. Sulla sinistraabbiamo l'accesso agli strumenti di base per manipolare la Scene (Scena) e gli oggetti che la compongono. Sulla destra invece sono presenti altri bottoni che danno l'accesso a Unity Collaborate, Unity Cloud Services, al proprio account personale, oltre a un menù di visibilità dei livelli e un altro menù del Layout che

permette di manipolare e salvare una configurazione diversa per le varie finestre che compongono l'Editor.

(B) La finestra **Hierarchy** è una rappresentazione testuale e gerarchica di ogni GameObject che compone la Scena. Ogni elemento nella Scena ha una voce nella gerarchia, quindi le due finestre sono legate intrinsecamente. La gerarchia rivela la struttura che lega i vari GameObject tra loro.

(C) La finestra **Game** simula come apparirà il gioco finale renderizzato attraverso la Camera presente nella Scena. Facendo click sul pulsante Play inizierà la riproduzione in modalità GamePlay della scena.

(D) La finestra **Scene** consente di navigare visivamente e modificare la scena. La vista Scena può mostrare una prospettiva 3D o 2D, a seconda del tipo di progetto su cui stai lavorando.

(E) La finestra **Inspector** consente di visualizzare e modificare tutte le proprietà del GameObject attualmente selezionato. Poiché diversi tipi di GameObject hanno diversi set di proprietà, il layout e il contenuto di questa finestra cambia ogni volta che selezioni un GameObjectdifferente.

(F) La finestra **Project** mostra la tua libreria di risorse disponibili per l'uso nel tuo progetto. Quando importi gli asset nel tuo progetto, vengono visualizzati qui.

Nell'interfaccia di Unity è presente di default anche un'altra finestra non meno importante: la finestra **Console**.[10]

[9]https://docs.unity3d.com/Manual/UsingTheEditor.html
[10]https://docs.unity3d.com/Manual/Console.html

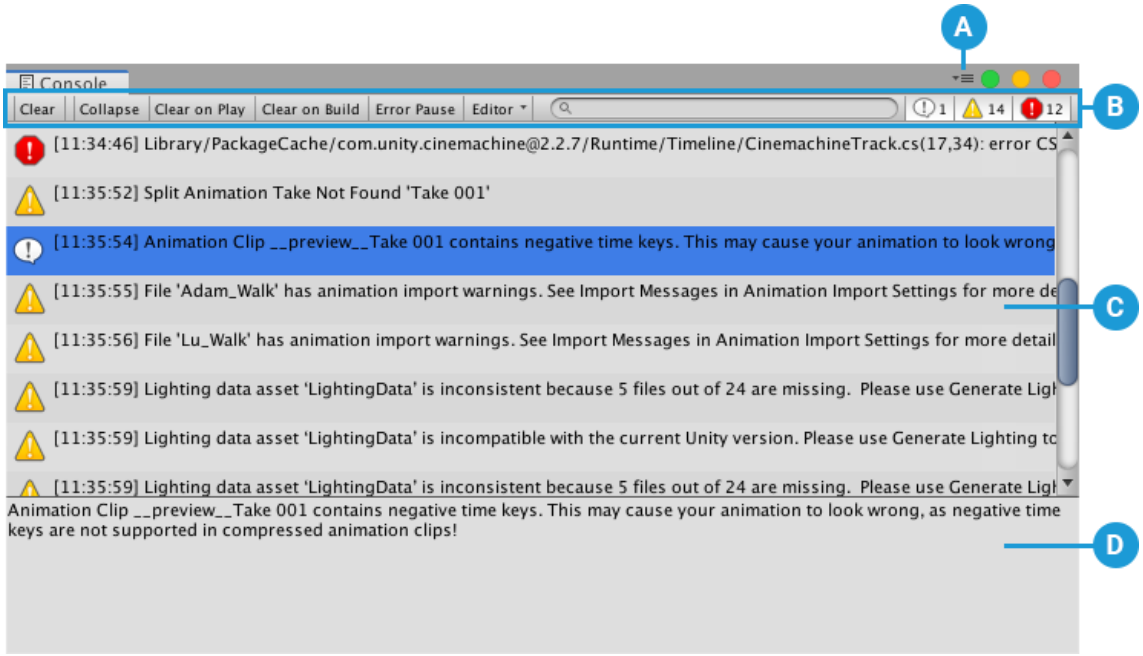


Figura 6: Finestra Console.

La finestra della Console mostra errori, avvisi e altri messaggi generati da Unity.

(A) Il menu della finestra della console dispone di opzioni per l'apertura dei file di registro, il controllo della quantità di ogni messaggio visibile nell'elenco e l'impostazione delle opzioni di analisi dello stack.

(B) La barra degli strumenti della Console dispone di opzioni per controllare la modalità di visualizzazione dei messaggi e per cercare e filtrare i messaggi.

(C) L'elenco della console visualizza una voce per ogni messaggio registrato. Selezionando un messaggio possiamo visualizzarne l'intero testo nell'area dei dettagli.

(D) L'area dei dettagli mostra il testo completo del messaggio selezionato.

2.1.2 Linguaggio di programmazione C#

Il linguaggio di programmazione utilizzato nell'ambiente di sviluppo di Unity è C# (C Sharp). C# è un linguaggio di programmazione orientato agli oggetti sviluppato da Microsoft all'interno della piattaforma .NET e successivamente approvato come standard internazionale dall'ECMA nel 2002 e dall'ISO nel 2003.

È stato progettato da Anders Hejlsberg, e il suo Il team di sviluppo è attualmente guidato da MadsTorgersen. La versione più recente è la 9.0, rilasciata nel 2020 in .NET 5.0 e inclusa in Visual Studio 2019 versione 16.8.

Lo standard Ecma elenca questi obiettivi di progettazione per C #:[11]

- Il linguaggio è concepito per essere un linguaggio di programmazione semplice, moderno, generico e orientato agli oggetti .
- Il linguaggio e le sue implementazioni dovrebbero fornire supporto per i principi di ingegneria del software come il controllo del tipo forte, il controllo dei limiti dell'array, il rilevamento dei tentativi di utilizzare variabili non inizializzate e la raccolta automatica dei rifiuti.
- Il linguaggio è destinato allo sviluppo di componenti software adatti per la distribuzione su molteplici ambienti.
- La portabilità è molto importante per il codice sorgente e i programmatori, specialmente quelli che hanno già familiarità con C e C ++ .
- Il supporto per l'internazionalizzazione è molto importante.
- C # è concepito per essere adatto alla scrittura di applicazioni sia per sistemi host che embedded, da quelli molto grandi che utilizzano sistemi operativi sofisticati, fino a quelli molto piccoli con funzioni dedicate.
- Sebbene le applicazioni C # siano destinate ad essere parsimoniose per quanto riguarda i requisiti di memoria e potenza di elaborazione , il linguaggio non è pensato per competere direttamente in quanto a prestazioni con il linguaggio C o assembly.

[11]https://en.wikipedia.org/wiki/C_Sharp_(programming_language)

Per quanto riguarda il sistema di scripting in Unity, schematizzando, possiamo illustrarlo in questo modo[12]: ogni oggetto nella scena dell'Editor di Unity è un *GameObject*: camere, luci, personaggi, oggetti 3D, effetti speciali, ecc.

Ogni *GameObject* può avere delle proprietà che vanno a definirne le caratteristiche. Possiamo assegnare delle proprietà ai *GameObject* tramite l'aggiunta di *Components*. Questi ultimi definiscono il comportamento dei vari *GameObject*: per esempio nella creazione di una fonte di luce andremo a creare un *GameObject* che avrà incorporato un *Component*Light che emette effettivamente la luce nella scena. Oppure, aggiungendo a un solido tridimensionale il *Component*RigidBody implementeremo la fisica sul*GameObject* in questione, rendendo quest'ultimo soggetto alla forza di gravità.

I *Components* hanno numerose proprietà editabili (variabili), che possono essere modificate nella finestra Inspector nell'Editor di Unity.

Per avere però un maggior controllo sulle proprietà dei *GameObject* o per produrre dei comportamenti più complessi o raffinati dovremmo utilizzare il sistema di scripting. Per implementare una propria logica dentro il programma infatti dovremmo affidarci alla creazione di Scripting *Components*(file di codice) che ci permetteranno di avere vasta libertà e ampio controllo sul comportamento da assegnare ai vari oggetti.

2.2 Librerie commerciali a pagamento

Per lavorare su progetti di Realtà Aumentata in Unity dobbiamo usufruire di alcuni kit di sviluppo specifici per questo ambito applicativo.

Sono disponibili molteplici librerie per lo sviluppo di software che si avvalgono di funzionalità in Realtà Aumentata, ognuna con i suoi punti di forza e le sue debolezze.

2.2.1 Vuforia

Vuforia è una delle società che da maggior tempo sono sul mercato della Realtà Aumentata. Dopo la sua acquisizione nel 2015 da parte di PTC Inc., Vuforia ha ampliato la sua linea di strumenti orientati all'AR. Questi strumenti ora includono

[12]https://unity.com/how-to/programming-unity

prodotti come *Vuforia Engine* e *Vuforia Studio*, entrambi utilizzati nello sviluppo di applicazioni AR.

L'attuale nona versione di *Vuforia Engine* presenta molte funzionalità:[13]

- *Model Targets*: consente alle app create utilizzando Vuforia Engine di riconoscere e tracciare oggetti particolari nel mondo reale in base alla forma dell'oggetto.

- *Area Targets*: è una funzionalità di tracciamento dell'ambiente che consente di tracciare e "aumentare" aree e spazi. È possibile utilizzare una scansione 3D come modello accurato dello spazio per creare un database dei dispositivi target dell'area, fornendo facilmente modelli di realtà aumentata agli oggetti fissi nell'ambiente di scansione.

- *Image Targets*: i target delle immagini rappresentano immagini che Vuforia Engine è in grado di rilevare e tracciare. Il motore rileva e tiene traccia dell'immagine confrontando le caratteristiche naturali estratte dall'immagine della telecamera con un database di risorse conosciute.

- *Multi target*: è una raccolta di più target d'immagine combinati in una disposizione geometrica definita come ad esempio una scatola.

- *Object Target*: I target degli oggetti vengono creati mediante la scansione di un oggetto. Sono una buona opzione per giocattoli e altri prodotti con ricchi dettagli di superficie e una forma definita.

- Simulazione Play Mode.

Dal 2017, Vuforia Engine è direttamente integrato con Unity3D, rendendo più semplice la creazione di progetti AR direttamente all'interno del motore di gioco (uno dei pregi di sviluppare con questa libreria è il fatto che per testare la scena nella modalità PlayMode in Unity possiamo anche utilizzare una webcam, senza quindi essere costretti a esportare il progetto sul dispositivo mobile per provare l'applicazione).

Si può utilizzare Vuforia Engine con iOS tramite XCode e con Android tramite Vuforia

[13] https://engine.vuforia.com/content/vuforia/en/features.html

Android SDK e Android Studio.

Vuforia Engine viene utilizzato principalmente con il motore di gioco Unity3D, quindi puoi trovare un'ampia documentazione di sviluppo sul forum ufficiale di Unity3D e sul sito Web di Vuforia.

Vuforia Studio, d'altra parte, è un'applicazione autonoma utilizzata per la creazione di esperienze AR senza avere necessariamente estese capacità di programmazione o progettazione.[14]

Pro:

- Sia le immagini che le forme possono fungere da marker per le esperienze AR, permettendoti di creare interazioni AR onnipresenti basate su tecniche di deep learning e modalità avanzate di riconoscimento delle forme.
- Ottima integrazione con Unity3D.
- Sono supportati diversi linguaggi di programmazione: C ++, Java, Objective-C e .NET attraverso l'estensione Unity3D.

Contro:

- Il prezzo è più alto rispetto ad altri strumenti AR. L'abbonamento è di $ 42 / al mese per la versione Basic; e di $99 / al mese per la versione Cloud. Tuttavia, è possibile sviluppare un progetto gratuitamente fino alla distribuzione dell'app.
- Nessun supporto per Unreal Engine 4.

2.2.2 Wikitude SDK

Wikitude SDK per Unity è un plugin per Unity3D, che aggiunge funzionalità di realtà aumentata. È ottimizzato per il rilevamento e il tracciamento di immagini e oggetti da utilizzare in esperienze di realtà aumentata o accanto a framework nativi per la realtà aumentata come ARCore, ARKit, AR Foundation o come strumento autonomo per alimentare un'esperienza AR.

[14] https://www.ptc.com/it/products/augmented-reality/vuforia-studio

Wikitude SDK per Unity fornisce le seguenti funzionalità:[15]

- Tracciamento delle immagini sul dispositivo: Il tracker analizza l'immagine della telecamera dal vivo e rileva i target memorizzati nella sua raccolta target associata.
- Tracciamento di oggetti cilindrici.
- Tracciamento di immagini da un server cloud.
- Tracciamento di più immagini contemporaneamente.
- Tracciamento di oggetti e scene**:** estende le capacità di Wikitude SDK di riconoscere e tracciare oggetti arbitrari per esperienze di realtà aumentata.Il motore di riconoscimento degli oggetti di Wikitude SDK può anche essere usato per riconoscere strutture più grandi che vanno oltre gli oggetti di dimensioni di una tabella, ad esempio stanze, facciate di edifici, piazze e cortili.
- Tracciamento di più oggetti contemporaneamente.
- Combina più tracker dall'alto in un'unica esperienza AR:È possibile eseguire contemporaneamente più istanze di tracking, rendendo così possibile combinare target di immagini, oggetti e cilindri nella stessa esperienza di AR.

Wikitude Studio Editor

Studio Editor è l'ultima piattaforma di Wikitude per la creazione di esperienze di realtà aumentata in pochi click. Editor consente agli sviluppatori e ai non sviluppatori di creare, gestire e pubblicare progetti AR basati su obiettivi e oggetti, il tutto in un'unica piattaforma senza possedere necessariamente competenze di programmazione.[16]

Funzionalità di Studio Editor:

- Basata sul Web: lavora in modo rapido ed efficiente senza l'installazione di software desktop.
- Facile da usare: non devi essere un esperto della programmazione per utilizzare Studio Editor.

[15]https://www.wikitude.com/external/doc/expertedition/#key-features
[16]https://www.wikitude.com/products/studio/

- Riconoscimento delle immagini: lavora con la tecnologia proprietaria integrata di Wikitude per il riconoscimento e il tracciamento delle immagini. Carica le immagini target nell'editor insieme alla sovrapposizione digitale desiderata
- Riconoscimento di oggetti e scene: Uno dei principali miglioramenti del nuovo Studio Editor è la funzionalità di riconoscimento di oggetti e scene, basata su SLAM (Simultaneouslocalization and mapping). L'interfaccia consente di creare una mappa dei punti 3D del target semplicemente filmandolo oscattandone delle foto.

Pro:

- Tecnologia affidabile.
- Molte funzioni disponibili.
- Supporto per il riconoscimento Cloud.

Contro:

- Prezzo un po' alto per la licenza di pubblicazione: la quota di acquisto una tantum per la versione PRO 3D (senza aggiornamenti della SDK) è 2490€, mentre la sottoscrizione della versione Cloud è 4490€ all'anno.[17]

2.3 AR Foundation

AR Foundation è un framework creato appositamente per lo sviluppo di applicazioni diRealtà Aumentata che consente di creare esperienze interattive e immersive con l'opportunità di distribuirle su molteplici piattaforme quali smartphone e visori indossabili[18]. Questa libreria nasce dalla volontà di Unity di fornire un'unica API che supporti le funzionalità di base di ARCore e ARKit.

[17]https://www.wikitude.com/store/
[18]https://unity.com/unity/features/arfoundation

ARCore è la libreria sviluppata da Google per la realizzazione di applicazioni in realtà aumentata su Android, mentre ARKit è la controparte sviluppatada Apple relativamente ai dispositivi iPhone e iPad. Entrambe implementano le stesse funzionalità di base per la realtà aumentata con alcune piccole differenze specifiche.

Uno dei vantaggi principali nell'utilizzo ARFoundation sta nella possibilità di sviluppare un unico progetto e di esportarlo senza modifiche a livello di codice su dispositivi di entrambe le piattaforme (Android e iOS). Ad ogni modo, ARFoundation non fornisce autonomamente funzionalità per la realtà aumentata ma un'interfaccia multipiattaforma unificata per lo sviluppo in Unity, andrà quindi installata insieme a un pacchetto specifico per la piattaforma di riferimento:[19]

- ARCore XR Plugin per Android.
- ARKit XR Plugin per iOS.
- Magic Leap XR Plugin per Magic Leap.
- Windows XR Plugin per HoloLens.

ARFoundation supporta le seguenti funzionalità:

- *Device tracking*: traccia la posizione e l'orientamento del dispositivo nello spazio fisico.
- *Planedetection*: rileva superfici orizzontali e verticali.
- *Point clouds*: conosciute anche come punti di interesse.
- *Anchor*: una posizione e un orientamento arbitrari di cui il dispositivo tiene traccia.
- *Light estimation*: esegue una stima per la temperatura media del colore e la luminosità nello spazio fisico.
- *Environment probe*: una tecnica per riflettere su un oggetto AR immagini provenienti dall'ambiente fisico.
- *Face tracking*: rileva e traccia i volti umani.
- *2D image tracking*: rileva e traccia le immagini 2D.

[19] https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@4.1/manual/index.html

- *3D object tracking*: rileva e traccia le immagini 2D.
- *Meshing*: genera mesh triangolari che corrispondono allo spazio fisico.
- *Body tracking*: rappresentazioni 2D e 3D di esseri umani riconosciuti nello spazio fisico.
- *Colaborativeparticipants*: traccia la posizione e l'orientamento di altri dispositivi in un'esperienza AR condivisa.
- *Raycast*: ricerca i dintorni fisici dei piani rilevati e i punti caratteristici.
- *Occlusion*: consente l'occlusione del contenuto virtuale in base alla profondità ambientale rilevata (occlusione ambientale) o alla profondità umana rilevata (occlusione umana).

2.3.1 Architettura di AR Foundation

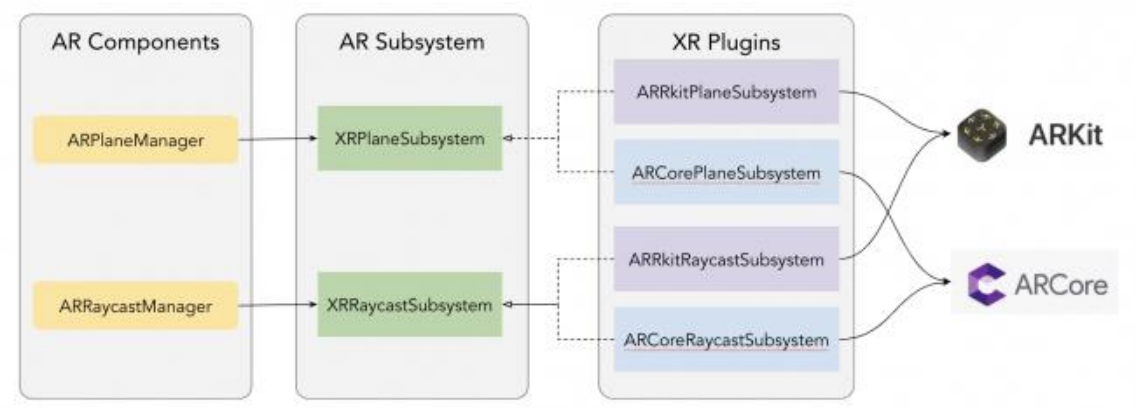


Figura 7: Architettura di AR Foundation

Ci sono due parti fondamentali dell'architettura di AR Foundation:[20]

- AR Subsystem
- AR Components

AR Subsystem è il ponte tra Unity e i moduli SDK nativi per la Realà Aumentata. Ogni Subsystem fornisce funzionalità diverse, ad esempio *ImageTrackingSubSystem* rileva le immagini e *RaycastSubSystem*permette di "lanciare raggi" all'interno della scena AR.

[20]https://www.raywenderlich.com/14808876-ar-foundation-in-unity-getting-started

AR Subsystem definiscele interfacce dei metodi ma non i dettagli di implementazione. I dettagli di implementazione effettivi sono scritti in diversi plugin XR (Extended Reality) specifici per ildispositivo, come il plug-in ARKit XR per iOS e il plug-in ARCore XR per Android. Questi plug-in implementano i metodi definiti in AR Subsystem utilizzando le funzioni native dei vari SDK.

I componenti AR invece sono molto più intuitivi per gli sviluppatori. Ecco alcuni componenti che incontreremo:

ARSession : per il ciclo di vita dell'AR

ARPlaneManager : per il rilevamento della superficie

ARRaycastManager : per rilevare il tocco dell'utente nell'ambiente AR

3 Workflow

Avendo scelto come libreria di sviluppo AR Foundation per sviluppare un'applicazione di realtà aumentata dobbiamo innanzitutto predisporre a questo scopo l'ambiente di Unity.

Come prima cosa apriamo l'Unity Hub, apriamo il menù Installs e selezionando l'Icona con i tre puntini relativa alla versione di Unity che stiamo usando facciamo click su AddModules. Aggiungiamo dunque i moduli per lo sviluppo sulle piattaforme su cui intendiamo lavorare: Andorid Build Support e/o iOS Build Support.

Inoltre, una volta aprto l'Editor di Unity è necessario infatti installare il plug-in di AR Foundation, più uno tra ARCore (per Android) o ARKit (per iOS) a seconda della piattaforma di riferimento per cui vogliamo sviluppare.

Per fare questo dobbiamo andare su *Window>Package Manager>Packages: UnityRegistry* e cercare i pacchetti di AR Foundation e ARCore o ARKit, e fare click su *Install*.

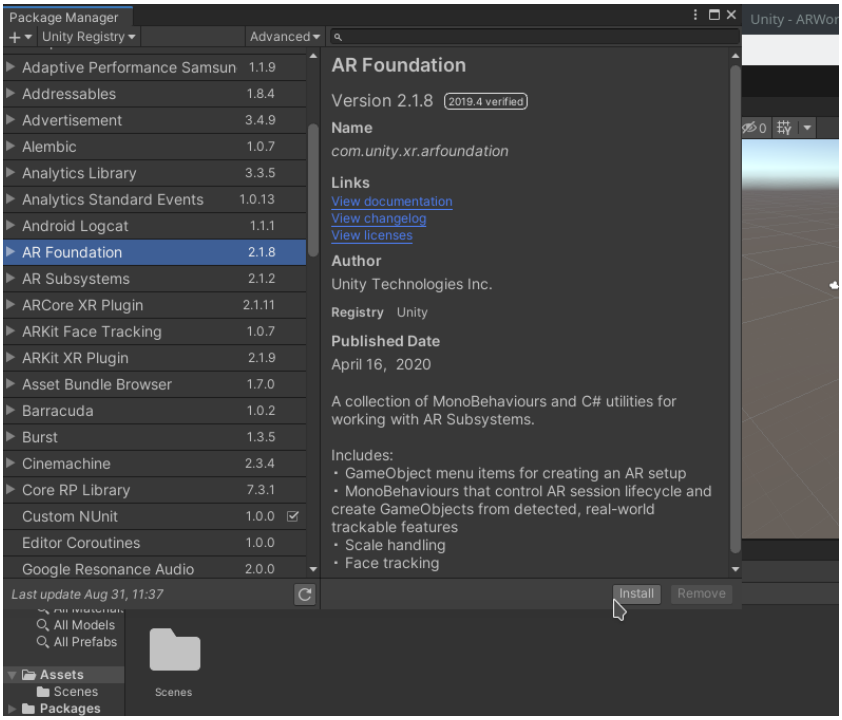


Figura 8: Schermata del Package Manager.

Se vogliamo decidere quale versione del pacchetto installare possiamo fare click sulla freccetta a sinistra del nome del pacchetto nella schermata del Package Manager e scegliere la versione tra quelle disponibili. Questa funzionalità è particolarmente utile nel caso le versioni più recenti dei pacchetti presentino delle incompatibilità, oppure per installare le versioni ancora in preview (facendo click su *Advanced>Advanced Project Settings* e mettendo la spunta sulla voce *Enable Preview Packages*) e testarne le funzionalità più recenti.

Dopo aver installato i pacchetti dobbiamo cambiare la piattaforma su cui andremo a esportare il progetto. Per fare questo andiamo su *File>Build Settings* e nella finestra Platforms andremo a selezionare la piattaforma su cui andrà a girare la nostra app (Android o iOS)infine faremo click su *Switch Platform*.

Inoltre,in *Edit>Project Settings>Player>Other Settings* dobbiamo andare a selezionare come Minimum API Level, la versione di Android 7.0 “Nougat” (API level 24) o una

successiva[21].

Se stiamo sviluppando per Android andiamo su *Edit>Project Settings>Player* e in *Other Settings* disabilitiamo la libreria Vulkan nella voce *Graphics API* o attiviamo la voce *Auto Graphics API*: così facendo la build che andremo a produrre includerà un set integrato di Graphics API e utilizzerà quella appropriata in fase di esecuzione per produrre uno scenario ottimale.[22]

C'è un ultimo passaggio che dobbiamo effettuare per rendere le impostazioni di Unity compatibili con le piattaforme mobile.

Andiamo su *Edit>Project Settings>XR Plug-in Management*. Qui selezioniamo il tab Android o iOS a seconda del nostro dispositivo di riferimento e poi, nella finestra Plug-in Providers spuntiamo la voce ARCore o ARKit, sempre secondo il nostro dispositivo.

Dopo aver completato anche questo passaggio abbiamo impostato le opzioni dell'Editor di Unity in modo da rendere l'ambiente compatibile con lo sviluppo di applicazioni di realtà aumentata su Android e iOS.

Per quanto riguarda il set-up realtivo allo sviluppo su un dispositivo iOS, una volta entrati in Unity (dopo aver installato il modulo relativo nell'Unity Hub), andiamo subito a selezionare la nostra piattaforma di riferimento (*File>Build Settings> iOS > Switch Platform*).

Dopodiché andiamo in *Edit>Project Settings>Player>Other Settings*mettiamo la spunta alla voce *RequiresARKit support*. Cambiamo la voce in *Architecture* da *Universal* a *ARM64* e infine cambiamo la voce *Target minimum iOS versio*n *11.0* che è la versione minima da iOS supportata da ARKit.

Poi installiamo il pluginARKit sempre andando in *Window>Package Manager*

Adesso viene finalmente il momento di elaborare la scena (Scene) inserendo i GameObject necessari per produrre un'esperienza di realtà aumentata.

Andiamo su *GameObject>XR>ARSessionOrigin* (oppure tasto destro nella finestra*Hierarchy>XR>AR Session Origin*).

[21]https://developers.google.com/ar/develop/unity-arf/quickstart-android

AR Session Originè un GameObject con attaccato uno Script di defalut (*ARSessionOrigin*) e con una Camera (*ARCamera*) come Child[23].

Lo scopo di *ARSessionOrigin* è trasformare gli elementi tracciabili, come le superfici e i marker, nella loro posizione, orientamento e scala finale nella scena Unity. Poiché i dispositivi AR forniscono i propri dati nello "spazio della sessione" (“session space”), che è uno spazio non in scala relativo all'inizio della sessione di Realtà Aumentata, *ARSessionOrigin* esegue la trasformazione dei relativi dati nello spazio diUnity.

Gli elementi tracciabili prodotti da un dispositivo AR, come i piani, sono forniti nello "spazio della sessione", rispetto al sistema di coordinate del dispositivo. Quando vengono istanziati in Unity come *GameObjects*, hanno anche uno “world space”. Per istanziarli nella posizione corretta, AR Foundation deve sapere dove dovrebbe essere l'origine della sessione nella scena Unity.[24]

L’altro componente indispensabile per creare un’esperienza in realtà aumentata è *ARSession*. Quest’ultimo controlla il ciclo di vita di un'esperienza in AR abilitando o disabilitando le funzionalità di realtà aumentata sulla piattaforma di destinazione. [25]

Con questi due componenti (ARSessionOrigin e ARSession) abbiamo messo a punto il set-updi base per una scenadi Realtà Aumentata.

Scena di esempio: Image Tracking

Se vogliamo ad esempio creare un’applicazione che inizializza delle animazioni su delle immagini marker quando il dispositivo mobile inquadra queste specifiche immagini possiamo effettuare i seguenti passaggi.

- Preparazione dell’ambiente di sviluppo (vedi sopra)

[22]https://docs.unity3d.com/2020.1/Documentation/Manual/GraphicsAPIs.html
[23] Un oggetto Child è un GameObject contenuto da un altro chiamato Parent. Questa operazione di collegamento dei GameObject tra di loro significa principalmente che l’oggetto figlio (child) subisce molte trasformazioni effettuate sul padre (parent) ma è utile anche a organizzare le voci degli oggetti presenti nella finestra Hierarchy, poiché quando il loro numero diventa elevato lavorare al progetto può diventare dispersivo.
[24]https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@4.1/manual/index.html
[25] Ibidem.

- Preparazione degli elementi di base per la scena in AR (vedi sopra)

Completati questi passaggi preliminari, andiamo nella finestra Asset, facciamo click col tasto destro e andiamo su *Create>XR>Reference Image Library*.

Selezionando quest'oggetto apparirà nell'Inspector la possibilità di aggiungere una o più immagini da utilizzare come target. Col pulsante *Add Image* possiamo aggiungere le immagini bidimensionali che verranno riconosciute dal dispositivo e innescheranno l'apparizione di un determinato oggetto o animazione.

Le immagini dovranno avere una quantità di dettagli (features points) minima affinché l'applicazione sia in grado di riconoscerla e identificarla.

C'è anche la possibilità di specificare le seguenti informazioni:

- *Dare un nome tramite una stringa di testo all'immagine che potrà essere utilizzato per capire quale immagine è stata identificata.*
- *Specificare le misure dell'immagine marker nel mondo reale in modo da facilitare il processo di riconoscimento.*
- *Possibilità di accedere alla texture di origine in RunTime. Di default questa impostazione è disabilitata per ridurre le dimensioni della build.*

Dopo aver inserito le immagini nella Library passiamo al *GameObject*ARSessionOrigin. Dopo averlo selezioniamo andiamo nella finestra dell'Inspector e premiamo sul pulsante *Add Component*. Si aprirà una piccola barra di ricerca nella quale andremo a cercare il componente *AR Tracked Image Manager*.[26]

Nell'Inspector possiamo vedere che questo componente ha, oltre uno script standard integrato che ne definisce la logica, alcuni parametri che possiamo andare a modificare e definire secondo le nostre necessità.

[26]https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@3.0/manual/tracked-image-manager.html#reference-library

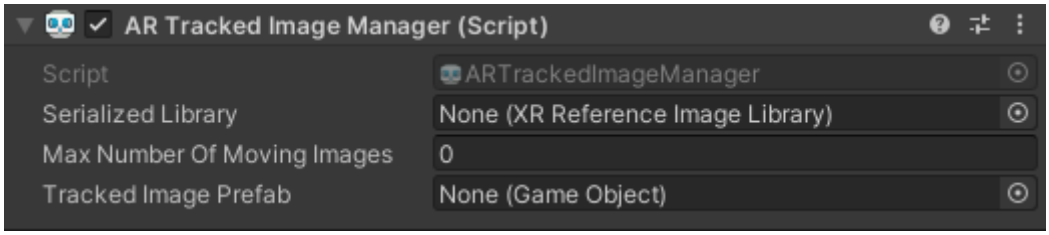


Figura 9: AR Tracked Image Manager.

La prima voce editabile è *Serialized Library* che corrisponde alla libreria di immagini che abbiamo trattato prima.

Infatti, per far funzionare la nostra app, dobbiamo andare a fare click su quel cerchio alla destra della casella vuota corrispondente alla voce e, dal menu a tendina che si aprirà, selezionare la nostra *Reference Image Library* (in alternativa possiamo direttamente trascinarla nella casella vuota). In questo modo indichiamo al componente *AR Tracked Image Library* quale immagine (o insieme di immagini) deve ricercare per il riconoscimento.

La seconda voce, *Max Numbers of Moving Images*, indica quante immagini l'applicazione può tracciare in contemporanea. Siccome il tracciamento di immagini in movimento può essere molto dispendioso in termini di risorse computazionali per alcuni dispositivi è opportuno, a seconda dei casi, mettere un limite.

La terza voce *Tracked Image Prefab* indica il Prefab[27] da inizializzare una volta che l'applicazione ha individuato e riconosciuto un'immagine target.*L'AR Tracked Image Manager* farà in modo che l'oggetto inizializzato sarà fornito del componente *AR Tracked Image*e sarà quindi tracciabile in tempo reale in modo da simulare un'integrazione del modello virtuale con l'ambiente reale.

Andremo quindi a trascinare in questa casella il Prefab che vogliamo appaia nella scena in AR quando l'immagine target inserita nella Library verrà identificata.

[27] I Prefabs ("Prefabbricati") sono un tipo speciale di componente che consente di salvare GameObject completamente configurati per un riutilizzo futuro. Queste risorse possono quindi essere condivise in scene differenti o anche in altri progetti senza dover essere nuovamente configurate.https://learn.unity.com/tutorial/prefabs-e

Effettuati questi passaggi potremmo direttamente passare alla fase di build per caricare l'applicazione sul nostro dispositivo mobile, in modo da testarne il funzionamento.

Per fare ciò, colleghiamo tramite USB il nostro dispositivo mobile alla macchina su cui stiamo sviluppando e dentro l'Editor di Unity andiamo su *File>Build Settings*, assicuriamoci di selezionare la piattaforma di riferimento nella finestra *Platforms* e di aggiungere la scena che abbiamo creato nella finestra *Scenes in Build*, facendo click sul pulsante *Add Open Scenes*, selezioniamo il dispositivo collegato dal menu a tendina alla voce *Run Device*, dopodiché clicchiamo su *Build And Run* per caricare l'applicazione sul dispositivo mobile.

Per testare una build su un dispositivo mobile dobbiamo abilitare la modalità di debug USB attivando le opzioni sviluppatore.

3.1 Gestione dei contenuti

3.1.1 Utilizzo di filmati come animazioni di stampe fotografiche

Tra le varie possibilità di utilizzo della Realtà Aumentata c'è anche quella che prevede la visualizzazione di video virtuali inseriti in un ambiente reale.

Una modalità applicativa di questo tipo lo abbiamo incontrato durante lo sviluppo di un'app per una mostra al Museo Madre di Napoli in cui contenuti video devono essere "innescati" quando l'utente inquadra delle immagini target poste in esposizione nelle sale del museo.

Per ottenere questo risultato abbiamo seguito i passaggi discussi nel paragrafo precedente (implementazione dell'*AR Tracked Image Library* per utilizzare le immagini 2D come trigger del video) e abbiamo aggiunto uno script ad hoc per permettere l'abbinamento di ogni immagine marker con un differente Prefab (per fare in modo che l'utente in visita alla mostra possa attivare i vari contenuti video in Realtà Aumentata in corrispondenza delle differenti immagini marker).[28]

Poi abbiamo impostato i prefab da inizializzare: abbiamo creato un oggetto 3D, un parallelepipedo quasi bidimensionale con le stesse dimensioni dell'immagine marker, in modo che l'oggetto in Realtà Aumentata appaia direttamente sovrapposto su quest'ultima.

Dopo aver creato l'oggetto 3D abbiamo aggiunto il component *Video Player* che fornisce la possibilità di allegare dei file video ai GameObjects.[29]

[28] Lo script di default di AR Tracked Image Manager infatti non prevede infatti di instanziareprefab multipli ognuno in corrispondenza di una relativa immagine target, ma solamente di instanziare lo stesso prefab attraverso differenti immagini.
[29]https://docs.unity3d.com/Manual/class-VideoPlayer.html

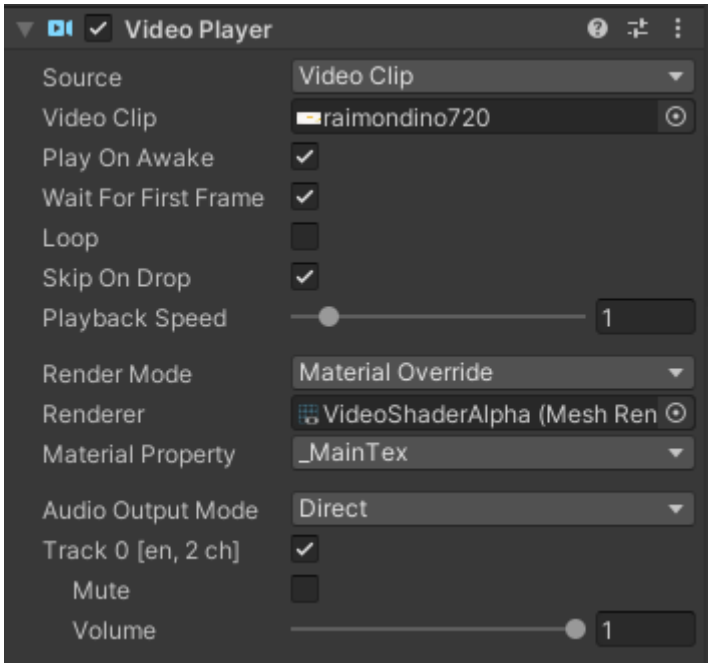


Figura 10: Video Player Component.

Nella casella *Video Clip* abbiamo inserito il file video che dovrà essere mostrato al visitatore in sede espositiva e che verrà renderizzatotramite una Texture di un materiale del GameObject a cui il componente è allegato (*Render Mode*: *MaterialOverride*).

Per garantire al video che appare sull'oggetto in AR la più alta fedeltà ai colori e alla qualità del file video originale, abbiamo utilizzato uno shader *Unlit* ovvero uno shader che rende il materiale dell'oggetto indipendente dalle fonti di luce presenti nella scena.

La luminosità quindi della texture del video sull'oggetto è quindi la stessa del video originale. L'immagine marker che fa partire il video costituisce sostanzialmente il primo fotogramma del video, in modo che l'avvio di quest'ultimo avvenga in un processo di sostanziale continuità tra l'ambiente fisico e l'oggetto virtuale. Questa impostazioneperò produceva un risultato che andava a inficiare l'esperienza di fruizione: il tono di bianco dell'immagine marker è infatti differente dal tono di bianco del video, quindi quando quest'ultimo veniva inizializzato si notava una discontinuità nei colori, nonostante l'immagine sia la stessa.

Per risolvere questo problema abbiamo creato una piccola animazione dell'oggetto con sopra il video in modo produrre una sorta di dissolvenza in entrata. Quando l'immagine

marker veniva riconosciuta, il GameObject con sopra il video veniva inizializzato ma il suo canale alfa (quello che regola la trasparenza) era settato inizialmente a zero e raggiungeva il valore di 100 in due secondi. In questo modo l'apparizione dell'oggetto con sopra il video avviene in maniera più "morbida" e fluida e la discrepanza dei due toni di bianco è mitigata dalla dissolvenza e non costituisce più un ostacolo percettivo.

3.1.2 Utilizzo di animazioni 3D

Durante la collaborazione ad un altro progetto, sempre riguardante la creazione di un'applicazione da utilizzare come supporto nella fruizione di una mostra, stavolta in un museo di storia naturale, ci siamo trovati a dover implementare delle animazioni in grafica vettoriale.

L'esperienza di fruizione dell'app da parte dei visitatori della mostra si può schematizzare così: in una serie di teche nelle sale del museo sono presenti i fossili di alcuni animali che componevano (o compongono) fauna locale.

In sede espositiva, vengono installate a fianco dei fossili delle immagini marker: utilizzando l'app su un dispositivo mobile e inquadrando queste immagini, il visitatore inizializza un oggetto 3D che mostra una ricostruzione animata dell'animale relativo al fossile in esposizione.

Inoltre, con alcuni sistemi di basici di Input (come fare tap sullo schermo dove è apparso l'oggetto 3D) possiamo produrre un'interattività, per quanto minimale, in modo da coinvolgere maggiormente l'utente nell'esperienza. Ad esempio: una volta che l'immagine target viene inquadrata ed è stata riconosciuta dall'app, l'oggetto 3D appare sullo schermo dello smartphone ma è inizialmente inanimato e per far partire l'animazione, l'utente deve toccare l'oggetto sullo schermo.

Per implementare animazioni tridimensionali in Unity dobbiamo innanzitutto utilizzare un file che sia compatibile con l'ambiente di sviluppo.

Unity può leggere file .fbx, .dae (Collada), .3ds, .dxf, e .obj.[30] Per questo progetto abbiamo utilizzato file con estensione .fbx creati con Cinema4D. Importando il file in Unity dobbiamo ricostruire il modello andando ricomporre materiali e texture relative

(che non vengono mantenute automaticamente nel processo di importazione).

Inoltre, dobbiamo anche a creare un Animator Controller abbinandoci l'animazione (realizzata precedentemente in Cinema4D) dell'oggetto corrispondente.

Una volta ricostruito il modello con texture e materiale e aggiunta l'animazione possiamo renderlo Prefab e aggiungerlo nella casella *Tracked Image Prefab*.

Abbiamo anche creato uno script per gestire l'interattività dell'utente (sistema di input touch) e la vibrazione che si attiva in certi momenti dell'animazione.

Le animazioni erano inizialmente previste con anche una serie di immagini in animazione frame-by-frame, che davano vita alla texture dell'animale rappresentato.

Con questo metodo l'apk che veniva creato in Unity aveva un peso molto elevato e considerando che ci dovevano essere circa quindici animazioni di questo tipo sarebbe stato difficile contenere le dimensioni dell'apk finale.

Per risolvere questo problema abbiamo individuato principalmente due vie: la prima è quella di sostituire le singole immagini dell'animazione con un video con un canale alfa che regola la trasparenza per lo sfondo, in modo da mostrare solo la figura dell'animale (questa soluzione però richiede di utilizzare un formato video particolare, .webm con video codec VP8, infatti non tutti i formati video supportano il canale alfa, e non tutti i formati con il canale alfa sono supportati da Unity[31]).

L'altra possibile soluzione era quella di trasformare le immagini .png in sprites 2D e realizzare l'animazione dentro l'Editor di Unity utilizzando il 2D Animation Package[32] (un pacchetto scaricabile dal Package Manager che permette di creare un animazione con uno "scheletro" e delle "ossa" manipolando direttamente una sprite 2D e riducendo quindi notevolmente il peso del progetto).

Alla fine abbiamo optato per la seconda opzione, anche per rendere tutto il processo produttivo concentrato principalmente su una sola suite per lo sviluppo, riducendo al minimo problema di incompatibilità e instabilità.

---

[30]https://docs.unity3d.com/Manual/3D-formats.html
[31] https://docs.unity3d.com/Manual/VideoTransparency.html
[32] https://blogs.unity3d.com/2018/11/09/getting-started-with-unitys-2d-animation-package/

4 Progetto VERO

Il prototipo che abbiamo sviluppato per il Progetto VERO incorpora alcune funzionalità che abbiamo già trattato e ne aggiunge di nuove. Abbiamo pensato a diverse modalità di fruizione per l'utente in modo da convogliare in un unico progetto, istanze di arricchimento e valorizzazione di un'importante opera d'arte (*La piazzetta dei Mosaici* realizzata da Venturino Venturi e ubicata nel Parco di Pinocchio a Collodi) e istanze ludiche/ interattive per coinvolgere adulti e bambini in una modalità di fruizione giocosa dei beni culturali.

4.1 Modalità Esplora

Il funzionamento di base della Modalità Esplora è sempre legato al riconoscimento di immagini target (in questo caso le "scene" raffigurate sui mosaici della Piazzetta nel Parco di Pinocchio) che attivano alcune animazioni tridimensionali che danno l'illusione che il mosaico si animi nello spazio.

In questo modo l'utente può esplorare liberamente i mosaici della pizzetta, attivare le varie animazioni in realtà aumentata e rivivere i momenti topici e i personaggi iconici de *Le Avventure di Pinocchio*.

Per aumentare la stabilità del tracking abbiamo implementato uno script custom che utilizza alcuni accorgimenti per migliorare l'esperienza complessiva. Nello script in questione è presente un contatore (1.5 secondi) che parte al riconoscimento dell'immagine. Se al termine del contatore l'immagine marker (in questo caso una "scena" dei mosaici) è ancora inquadrata attiviamo l'animazione corrispondente. Questa funzione serve a evitare che l'avviamento dell'animazione non avvenga in maniera casuale ma sia l'esito della volontà dell'utente.

Dopodiché aggiorniamo la posizione in base al movimento dell'utente. Inoltre, a seconda del contenuto da mostrare all'utente (la tipologia di animazione) abbiamo bloccato la rotazione dell'oggetto animato, in modo da cercare di produrre un'esperienza percettiva complessiva coerente, leggibile ed esteticamente soddisfacente.

4.2 Modalità Gioco

Oltre a questo, abbiamo pensato a un sistema di interazione in forma di gioco a quiz che metta alla prova l'utente sulla sua conoscenza della storia di Pinocchio.

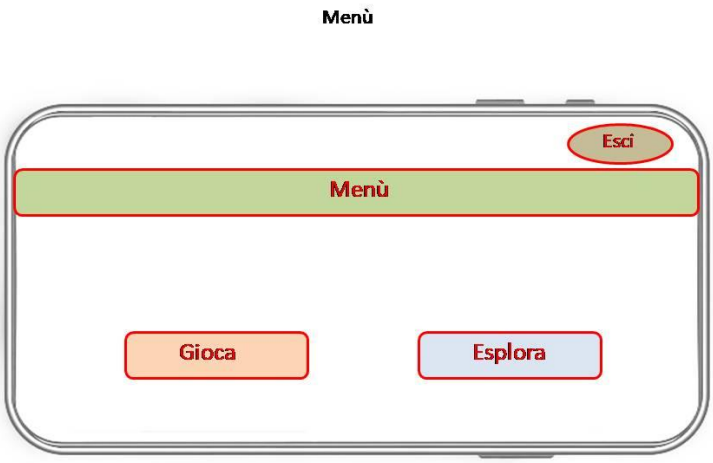
Il sistema gioco da svolgersi all'interno della Piazzetta dei Mosaici è articolato secondo due opzioni: la prima ("Quiz") consiste in una serie di domande poste all'utente sulla storia di Pinocchio e ogni risposta è (contenuta in) una scena del mosaico.

La seconda opzione, "Ricostruisci la storia", consiste nel proporre al visitatore di andare a inquadrare i vari mosaici (che non sono disposti secondo un ordine cronologico) secondo l'ordine di svolgimento degli avvenimenti della storia.

In entrambi i casi il funzionamento della modalità gioco avviene all'interno de *La Piazzetta dei Mosaici* avrà una struttura di base simile, possiamo immaginarne i passaggi principali in questo modo:

- Schermata iniziale che mostra il Logo dell'app.

- Un menù iniziale a schermo che permette di selezionare la modalità Gioca.



- Dopo aver scelto la modalità Gioca, partirà il feed della camera del dispositivo. Un piccolo messaggio a schermo avvertirà l'utente di fare tap sullo schermo (indicando un punto del pavimento della piazzetta) per piazzare un Avatar tridimensionale di Pinocchio che fornirà informazioni ai visitatori e spiegherà le regole del gioco (In questa fase ci saranno un *Planedetection* e un *ARraycastManager*attivi in modo che l'utente possa fare tap su un piano individuato dall'app e piazzare così l'avatar di Pinocchio).

Fai tap sullo schermo per piazzare l'avatar su un piano e iniziare il gioco

- Una volta inizializzato, l'avatar 3D di Pinocchio fornirà agli utenti informazioni di base sulla piazzetta dei mosaici e indicherà le varie modalità di gioco attraverso un'interfaccia volumetrica in world space a forma di nuvola dei fumetti. L'utente può interagire attraverso pulsanti.

Gioca 1

- Una volta selezionata la modalità di gioco a quiz, apparirà una domanda tramite un'interfaccia a schermo. Apparirà inoltre, sempre a schermo, un'indicazione riguardo la modalità di risposta: l'utente è invitato a spostarsi e a inquadrare i vari mosaici alla ricerca della risposta esatta (semplici frecce animate che suggeriscono l'idea di movimento)

Quiz 1

- Per rispondere l'utente dovrà inquadrare il mosaico che ritiene costituisca la riposta esatta. Per mostrare agli utenti le possibili opzioni di risposta, una volta inquadrato un feature point su un mosaico apparirà un piccolo indicatore (magari un'icona di Pinocchio), evidenziando così che quella è una possibile risposta. Toccando il puntatore l'utente confermerà la risposta.

Quiz 2 - Risposta

- Se la risposta è sbagliata al rilascio del touch l'icona diventerà rossa (e apparirà in sovrimpressione un messaggio con scritto "risposta errata, riprova!")

Quiz 2 – risposta errata

- Se invece la risposta è esatta il puntatore si illuminerà di verde e partirà l'animazione del mosaico. Dopodiché verrà posta una nuova domanda.

Quiz 3 – Risposta esatta

- La modalità "Ricostruisci la storia" utilizza la stessa dinamica di scelta attraverso le inquadrature delle "scene" mosaicate e lo stesso sistema di

valutazione delle risposte giuste o sbagliate, solo che invece di rispondere a delle domande, l'utente dovrà ricostruire cronologicamente le varie tappe della storia di Pinocchio. In altre parole, dovrà inquadrare una per una le varie "scene" nell'ordine giusto.

4.3 Deployment su piattaforme Android e iOS

Come abbiamo accennato in precedenza ci sono due finestre nelle quali possiamo cambiare le impostazioni di esportazione su dispositivi mobile:[33]

- *Player settings*: consente di configurare molteplici impostazioni di runtime per l'app.
- *Build settings*: consente di configurare i parametri del sistema di compilazione, la piattaforma di destinazione e finalmente di creare l'app.

Attraverso la finestra *Player Settings* (che si trova anche in *Edit>Project Settings>Player*) possiamo, cambiare il nome dell'app, assegnargli un'icona, impostare la modalità di fruizione standard (schermo verticale o orizzontale), scegliere la durata dello splash screen iniziale, ecc.
Oltre a dati estetici c'è la possibilità di settare una gran quantità di altri parametri[34]

[33]https://docs.unity3d.com/Manual/android-BuildProcess.html
[34]https://docs.unity3d.com/Manual/class-PlayerSettingsAndroid.html

come ad esempio,le API grafiche da utilizzare o, se esportiamo su Android, la versione minima del sistema Android su cui il programma è eseguibile (*Minimum API Level*).
Nella finestra *Build Settings* invece selezioniamo la piattaforma di destinazione, il dispositivo su cui eseguire l'app, le scene da inserire nella build e alcuni parametri tecnici come la compressione delle texture.

Una volta che abbiamo impostato tutti i parametri che ci interessano secondo le nostre esigenze possiamo finalmente esportare l'app sul nostro dispositivo (*File >Build Setting>Build and Run*).

Unity supporta due tipi di sistemi di building per Android: Gradle e Internal.[35]
Il primo sistema di compilazione utilizza Gradle per creare un APK o esportare un progetto in formato Gradle, che può quindi essere importato in Android Studio. Quando si seleziona questo sistema di compilazione, Unity esegue gli stessi passaggi del sistema di compilazione interno escludendo la compilazione delle risorse con AAPT, l'unione dei manifesti e l'esecuzione di DEX. Unity quindi genera il file build.gradle (insieme agli altri file di configurazione richiesti) e richiama l'eseguibile Gradle, passandogli il nome dell'attività e la directory di lavoro. Infine, l'APK è costruito da Gradle.

Il sistema di build interno invece crea un APK utilizzando le utilità Android SDK per creare e ottimizzare i pacchetti APK e OBB.

I passaggi coinvolti con la creazione per Android sono:

- Preparare e costruire gli Unity Assets.
- Compilazione di script.
- Elaborazione dei plug-in.
- Suddivisione delle risorse nelle parti che vanno all'APK e all'OBB, se è selezionato Split Application Binary.
- Creazione delle risorse Android utilizzando l'utilità AAPT (solo build interna).
- Generazione del manifest Android.

[35]https://docs.unity3d.com/Manual/android-BuildProcess.html

- L'unione del “library manifest” nel “Androidmanifest” (solo build interna).
- Compilazione del codice Java nel formato DalvikExecutable (DEX) (solo build interna).
- Creazione di IL2CPP libreria, se è selezionato IL2CPP Scripting Backend.
- Creazione e ottimizzazione dei pacchetti APK e OBB. Sistema di costruzione

4.4 Piattaforme di distribuzione digitale

4.4.1 Google Play Store

Per pubblicare un’applicazione sul Play Store dobbiamo creare un account sviluppatore e pagare la quota d’iscrizione che corrisponde ad un unico importo di 25 $.[36]

Per fare ciò dobbiamo andare sul portale Google Play Console e registrare un account su questa piattaforma.

Quando abbiamo eseguito questi passaggi e abbiamo accettato il contratto di distribuzione con gli sviluppatori per Android Market possiamo finalmente effettuare l’upload del file .apk della nostra applicazione.

Una volta che il file è stato inviato sul Play Store, Google effettua un controllo di routine prima della pubblicazione vera e proprio, per cui potrebbe passare qualche ora affinché l’applicazione risulti visibile sullo store online.

Se vogliamo caricare un’app Android a pagamento o se è presente qualche forma di monetizzazione nell’applicazione (ad esempio tramite acquisti in-app, abbonamenti, ecc.), abbiamo bisogno di un account per i commercianti su Google Centro Pagamenti, che può essere creato altrettanto velocemente. Andiamo nella sezione di Google Play Developer Console nella sezione “Account commerciante”, clicchiamo su “Configura l’account commerciante” e così verremo reindirizzati nella pagina per creare un profilo pagamenti, dove inserire tutti i dati relativi alla nostra attività (come ragione sociale dell’azienda, sede, ecc.).

[36]https://support.google.com/googleplay/android-developer/answer/6112435?hl=it

4.4.2 Apple App Store

Per creare un'app iOS e pubblicarla sull'App Store, dovremmo registrarci come Sviluppatore Apple. L'account cosa 99€ e deve essere rinnovato ogni anno. Entrare a far parte di questo programma ci consentirà anche di costruire un'app iOS e testarla sui nostri dispositivi, un passo cruciale nel processo di sviluppo dell'app.

Una volta esportata l'app definitiva possiamo inviarla all'App Store. Prima di inviare l'app per la revisione, dovremmo fornire una raccolta di informazioni (Icona, anteprima dell'app/screenshot) e metadati (il nome della tua app, la sua categoria, una descrizione dettagliata e parole chiave aggiuntive per l'ASO). Le prime 3 righe della descrizione sono le più importanti perché possono essere visualizzate dagli utenti senza dover espandere il testo. Il nome dell'account sviluppatore deve essere lo stesso del proprietario dell'app. È il nome che appare sotto la tua app nello Store.

Il processo di revisione impiega alcuni giorni, dopodiché Apple ti informerà quando la tua app sarà disponibile pubblicamente nello Store.

4.5 Deployment su occhiali dedicati

Anche se fino ad adesso non abbiamo potuto testare questa opzione, c'è la possibilità di distribuire le applicazioni di realtà aumentata sugli appositi visori, come ad esempio, gli Epson Moverio (smartglass binoculari dotati di lenti trasparenti e display OLED).

Epson infatti ha rilasciato un kit, il Moverio AR SDK[37], che permette agli sviluppatori di provare a sperimentare funzionalità di realtà aumentata (e/o di realtà virtuale) su alcuni dispositivi (in questo caso per i seguenti modelli Moverio: BT-300, BT-350, BT-2000, BT-2200).

Anche altri produttori di smartglass per la realtà aumentata hanno reso disponibili dei SDK per permettere agli sviluppatori di programmare sui loro dispositivi: uno di questi è Magic Leap.

Dopo aver effettuato la registrazione sul loro portale possiamo scaricare Magic Leap

[37]https://tech.moverio.epson.com/en/arsdk/sdk_download.html

Lab che dà accesso alle risorse per lo sviluppo sui visori di questa compagnia.[38]

Sicuramente uno dei vantaggi di questa tipologia di dispositivi è la loro immediatezza nell'esperienza. Infatti, a differenza di uno smartphone che interpone tra l'utente e l'esperienza del mondo in AR un piccolo schermo, avere dei visori binoculari aumenta l'immersione, il coinvolgimento e libera le mani avendo così la possibilità di implementare un sistema di input più profondo e intuitivo.

Per raggiungere un grado maggiore di interattività nella nostra app una delle possibili opzioni infatti è integrare, in un ambiente di realtà aumentata, un sistema di input più sofisticato e slegato dallo schermo dello smartphone. In quest'ottica una possibilità da esplorare per futuri sviluppi potrebbe essere quella di utilizzare in qualche progetto un sistema di input che riconosca i gesti delle mani (Hand Gesture).
In questo modo l'utente ha modo di interagire con l'ambiente di realtà aumentata senza dover passare dall'interfaccia dello schermo touch del dispositivo.
Inoltre, avendo a disposizione una serie di gesti possibili, aumentano le possibilità di "manipolare" l'ambiente AR (e gli oggetti contenuti in esso) e configurando nuove modalità di esperienza per l'utente.

Ci sono già dei progetti che si muovono in questa direzione. Manomotion ad esempio, è un plugin che incorpora alcune funzionalità simili:[39]

- *Hand Tracking*
- *Skeleton Recognition*
- *Advanced Gesture Control*
- *POI (Point of Interaction)*

Queste funzionalità sono molto interessanti perché aprono scenari inediti riguardo l'approccio dell'utente con l'ambiente interattivo in AR. Infatti, progettare un sistema di input basato sui gesti delle mani e svincolato dal piccolo schermo degli smartphone

[38]https://developer.magicleap.com/en-us/learn/guides/mltk-setup-unity

corrisponde a una sfida per gli sviluppatori e a una promessa di esperienze sempre più profonde e coinvolgenti, soprattutto se combinate con degli occhiali dedicati (come, ad esempio, gli Epson Moverio).

Eliminare l'intermediazione dello schermo dello smartphone tramite gli smartglasses e aggiungere un sistema di input articolato e stimolante costituiscono forse i prossimi passi da fare per aggiungere un altro grado di maturazione alla tecnologia della Realtà Aumentata.

[39]https://www.manomotion.com/products/

5 Bibliografia

Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., Hallaway, D.: *Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system*, Articolo in Rivista, Comput. Graph. **23**(6), 779–785 (1999).

Vlahakis, V., Ioannidis, N., John, K., Tsotros, M., Gounaris, M.: *Archeoguide: an augmented reality guide for archaeologicalsites*, Articolo in Rivista, Comput. Graph. Art Hist. Archaeol. **22**(5), 52–60 (2002).

5.1 Sitografia

http://avrlab.it/augmented-reality-for-the-enhancement-of-apulian-archaeological-areas/

http://avrlab.it/scienza-machinale-aumentata/

https://blogs.unity3d.com/2018/11/09/getting-started-with-unitys-2d-animation-package/

https://developer.magicleap.com/en-us/learn/guides/mltk-setup-unity

https://developers.google.com/ar/develop/unity-arf/quickstart-android

https://docs.unity3d.com/2020.1/Documentation/Manual/GraphicsAPIs.html

https://docs.unity3d.com/Manual/3D-formats.html

https://docs.unity3d.com/Manual/android-BuildProcess.html

https://docs.unity3d.com/Manual/android-BuildProcess.html

https://docs.unity3d.com/Manual/class-PlayerSettingsAndroid.html

https://docs.unity3d.com/Manual/class-VideoPlayer.html

https://docs.unity3d.com/Manual/Console.html

https://docs.unity3d.com/Manual/UsingTheEditor.html

https://docs.unity3d.com/Manual/VideoTransparency.html

https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@3.0/manual/tracked-image-manager.html#reference-library

https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@4.1/manual/index.html

https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@4.1/manual/index.html

https://en.wikipedia.org/wiki/C_Sharp_(programming_language)

https://engine.vuforia.com/content/vuforia/en/features.html

https://hiddenflorence.org/

https://support.google.com/googleplay/android-developer/answer/6112435?hl=it

https://tech.moverio.epson.com/en/arsdk/sdk_download.html

https://unity.com/dots

https://unity.com/how-to/programming-unity

https://unity.com/unity/features/arfoundation

https://www.dianamarques.com/portfolio/augmented-reality-app-skin-bones/

https://www.manomotion.com/products/

https://www.ptc.com/it/products/augmented-reality/vuforia-studio

https://www.raywenderlich.com/14808876-ar-foundation-in-unity-getting-started

https://www.terramosana.org/

https://www.wikitude.com/external/doc/expertedition/#key-features

https://www.wikitude.com/products/studio/

https://www.wikitude.com/store/

NOTA: Loghi ed immagini del Parco di Pinocchio appaiono per gentile concessione della Fondazione Nazionale Carlo Collodi, con la quale attivo un rapporto di collaborazione nell'ambito del progetto VERO.