RICONOSCIMENTO IMMAGINI

Design Requirement Specification Document

DIBRIS – Università di Genova. Scuola Politecnica, Corso di Ingegneria del Software 80154

**DATA – 16 Aprile 2018**

**VERSION: V1**

**Autori**

Edoardo Bernardi

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Versione | Data | Autori | Note |
| V1 | 16 aprile 2018 | Edoardo Bernardi | Prima versione |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Indice dei Contenuti

1 [Introduzione 3](#__RefHeading___Toc106_1667604106)

[1.1 Scopo del Documento 3](#__RefHeading___Toc108_1667604106)

[1.2 Descrizione Generale del Sistema 3](#__RefHeading___Toc110_1667604106)

[1.3 Riferimenti Bibliografici 3](#__RefHeading___Toc1851_1623032702)

[1.4 Definizioni e Acronimi 3](#__RefHeading___Toc112_1667604106)

[1.5 Overview del documento 4](#__RefHeading___Toc1853_1623032702)

[2 Descrizione del Progetto 5](#__RefHeading___Toc1855_1623032702)

[2.1 Introduzione 5](#__RefHeading___Toc1857_1623032702)

[2.2 Architettura del Sistema 5](#__RefHeading___Toc1859_1623032702)

[2.3 Vincoli e Assunzioni 5](#__RefHeading___Toc1861_1623032702)

[3 Interfacce 5](#__RefHeading___Toc1863_1623032702)

[4 Data Stores 7](#__RefHeading___Toc1865_1623032702)

[5 Use Cases 7](#__RefHeading___Toc1865_1623032702)

[6 Structural Design 8](#__RefHeading___Toc1867_1623032702)

[6.1 Class Diagram 8](#__RefHeading___Toc1869_1623032702)

[6.1.1 Descrizione delle classi 8](#__RefHeading___Toc1871_1623032702)

[6.2 Object Diagram 10](#__RefHeading___Toc1873_1623032702)

[7 Dynamic Model 12](#__RefHeading___Toc1875_1623032702)

7[.1 Requisiti 12](#__RefHeading___Toc1877_1623032702)

# Introduzione

## Scopo del Documento

Il documento sottoposto ha lo scopo di descrivere l’architettura del sistema software atto al riconosciemento di immagini richiesto da Untold Games S.r.l.. Questo è destinato a progettisti e sviluppatori per facilitare la fase realizzativa del software stesso e a tester per la fase di verifica.

## Descrizione Generale del Sistema

Il sistema descritto è una applicazione mobile che, utilizzando TF, sia in grado di riconoscere un determinato oggetto ed identificarlo su schermo sfruttando l’ AR. L'applicazione non deve riconoscere oggetti generici ma specifici.

## Riferimenti Bibliografici

* Sito web di TF con dettaglio sul riconoscimento di oggetti: <https://www.tensorflow.org/tutorials/image_recognition>
* Informazioni su object detection e reti neurali

<http://www.wolfib.com/Image-Recognition-Intro-Part-1/>

<http://cv-tricks.com/tensorflow-tutorial/training-convolutional-neural-network-for-image-classification/>

* Repository github con tutorial base di object detection: <https://github.com/tensorflow/models/tree/master/research/object_detection>
* Scelta modello per object detection

<https://medium.com/@WuStangDan/step-by-step-tensorflow-object-detection-api-tutorial-part-1-selecting-a-model-a02b6aabe39e>

* Discussione generale

https://towardsdatascience.com/is-google-tensorflow-object-detection-api-the-easiest-way-to-implement-image-recognition-a8bd1f500ea0

## Definizioni e Acronimi

|  |  |
| --- | --- |
| **Acronimo-Nome** | **Definizione** |
| SE | Software Engineering |
| TF | TensorFlow |
| AR | Augmented Reality: aggiunta di informazioni multimediali alla realtà per mezzo di dispositivi dotati di strumenti di visione. |

## 

## Overview del documento

Il documento presenta sette parti differenti:

1. Introduzione: Mostra lo scopo del documento e una descrizione del sistema software. Si trovano in aggiunta link utili ed acronimi utilizzati, si concolude con questo paragrafo.
2. Descrizione del progetto: Viene fornita una descrizione ad alto livello dell’applicazione descrivendone architettura e vincoli da rispettare.
3. Interfacce: Viene mostrata l’interfaccia del sistema.
4. Data Stores: Viene descritto come avviene l’eventuale memorizzazione dei dati.
5. Use Case: Fornisce una descrizione delle funzioni svolte dai diversi utenti del sistema.
6. Structural Design: Fornisce una descrizione nel dettaglio del sistema per mezzo di diagrammi strutturali quali Class Diagram ed Object Diagram.
7. Dynamic model: Viene descritto il comportamento dinamico del sistema ovvero come si comporta il sitema in reazione ad eventi esterni (inquadra, blocco if per riconoscimento e box)

# Descrizione del Progetto

## Introduzione

Il processo di design adottato per il progetto è di tipo object oriented ed un’architettura atta ad un’implementazione su dispositivo mobile. Per i diagrammi UML è stato utilizzato il software DIA.

## Architettura del Sistema

Il sistema è costituito da una singola applicazione mobile, che si interfaccia con la fotocamera del dispositivo, utilizzata per il riconoscimento di un oggetto ben preciso. Venendo sfruttata una rete precedentemente addestrata non è necessaria la memorizzazione dati su database.

## Vincoli e Assunzioni

Trattandosi di un’applicazione che necessita che il riconoscimento di oggetti avvenga in tempo reale è necessario un compromesso tra precisione e rapidità. Da un punto di vista del design non ha influenza ma in fase di addestramento della rete neurale dovrà essere scelto un modello opportuno.

Il sistema deve essere implementato su dispositivi con sistema operativo Android.

Il sistema può essere implementato su dispositivi con sistema operativo IOS.

# Interfacce

## BaseInterface

Si tratta dell’interfaccia di base che si ha all’apertura dell’applicazione. All’utente viene mostrato a schermo ciò che viene inquadrato dalla fotocamera.



*Figura 3.1: L’applicazione restituisce a schermo l’ambiente circostante.*

## ObjectInterface

Si tratta dell’interfaccia che si ha quando l’applicazione riconosce l’oggetto per il quale è stato eseguito l’addestramento della rete neurale. È una variazione dell’interfaccia precedente: a schermo viene sempre restituita l’immagine acquisita dalla fotocamera ma attorno all’oggetto compare un riquadro colorato per evidenziarlo; inoltre viene mostrata la percentuale di affidabilità con la quale è avvenuto il riconoscimento.



*Figura 3.2: L’applicazione evidenzia l’oggetto riconosciuto e l’affidabilità.*

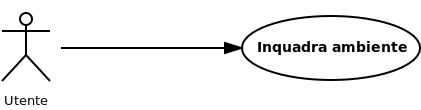
# Data Stores

Non è necessaria la memorizzazione dati in quanto per il riconoscimento di un oggetto specifico viene utilizzata una rete neurale precedentemente addestrata.

# Use Cases

Il sistema prevede la presenza di un solo utente, un utente generico che necessita di riconoscere l’oggetto specifico per cui è stata addestrata la rete neurale.

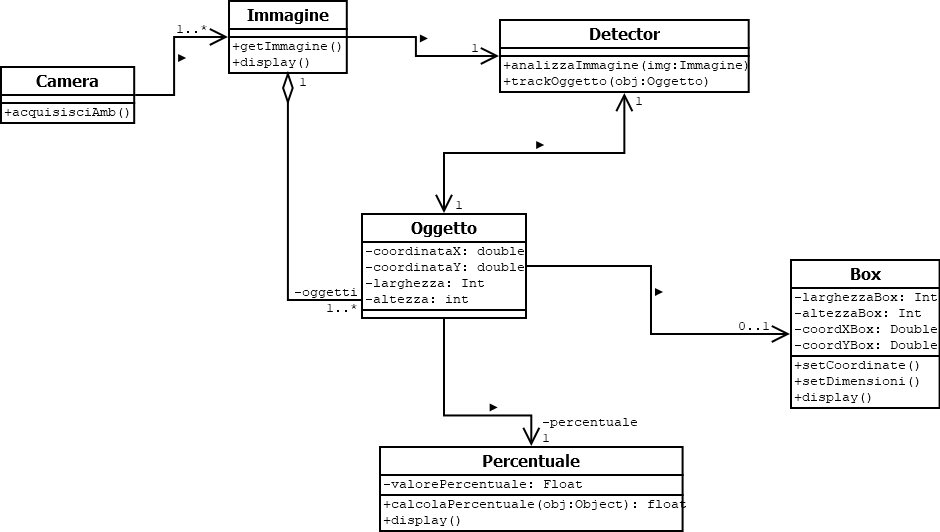
L’utente in questione deve solamente inquadrare l’ambiente circostante con la fotocamera del dispositivo, sarà quindi l’applicazione stessa a riconoscere l’oggetto specifico se esso venisse inquadrato e a porre su schermo il riquadro e la percentuale di affidabilità.



*Figura 5: L’utente inquadra l’ambiente e l’applicazione effettua un eventuale riconoscimento.*

# Structural Design

## Class Diagram



*Figura 6.1: Class Diagram*

### Camera

La classe Camera rappresenta la fotocamera fisica del dispositivo.

*Metodi:*

* *acquisisciAmb():* l’acquisizione dell’ambiente da parte della fotocamera del dispositivo.

### Immagine

La fotocamera acquisice l’ambiente, ogni frame di ciò che viene acquisito costituisce un’immagine differente. La classe immagine rappresenta quindi un frame di ciò che acquisisce la fotocamera.

*Metodi:*

* *getImmagine():* acquisizione di un’immagine, un frame dalla fotocamera.
* *display():* mostra a schermo l’immagine acquisita dalla fotocamera.

### Detector

La classe Detector rappresenta l’entità che si occupa di eseguire l’analisi dell’immagine identificando i differenti oggetti e di eseguire il tracking per facilitare l’identificazione dell’oggetto nei frame successivi.

*Metodi:*

* *analizzaImmagine(img:Immagine):* esegue l’analisi dell’immagine identificando i differenti oggetti al suo interno.
* *trackOggetto():* metodo che facilita il tracking dell’oggetto nei frame successivi acquisiti dalla fotocamera.

### Oggetto

La classe Oggetto rappresenta invece uno degli oggetti che vengono inquadrati dalla fotocamera e che costituiscono l’immagine acquisita.

*Attributi:*

* *coordinataX : Double* indica la coordinata x dell’oggetto analizzato.
* *coordinataY : Double* indica la coordinata y dell’oggetto analizzato.
* *larghezza : Int* indica la larghezza dell’oggetto analizzato.
* *altezza : Int* indica ll’altezza dell’oggetto analizzato.

I vincoli sulle coordinate e dimensioni verranno determinati in fase implementativa.

### Percentuale

La classe Percentuale rappresenta invece la percentuale di affidabilità con cui avviene il riconoscimento di un oggetto: ogni oggetto ha infatti una percentuale che indica quanto esso “potrebbe essere” l’oggetto per cui è stata addestrata la rete neurale; quando questa percentuale è sopra un certo valore allora l’oggetto verrà riconosciuto come l’oggetto specifico.

*Attributi:*

* *valorePercentuale: Float* indica il valore percentuale di riconoscimento. Tale valore è compreso tra 0 e 1.

*Metodi:*

* *calcolaPercentuale(obj: Oggetto) : float:* il metodo esegue il calcolo della percentuale di affidabilità con cui un oggetto viene riconosciuto come quello specifico per cui la rete neurale è stata addestrata. Viene passato come parametro un oggetto di tipo Oggetto e viene restituito un Float ovvero la percentuale (compresa tra 0 ed 1).
* *display():* metodo per eseguire il display della percentuale. Come verrà mostrato nel modello dinamico, il display delle informazioni avviene solo se la percentuale di riconoscimento è sopra un certo valore percentuale.

### Box

La classe Box rappresenta il riquadro che circonda l’oggetto specifico.

*Attributi:*

* *Larghezza: Int* larghezza del riquadro.
* *Altezza: Int* altezza del riquadro.
* *coordXBox: Double* coordinata x del centro del riquadro.
* *coordYBox: Double* coordinata y del centro del riquadro.

*Metodi:*

* *display(): ):* metodo per eseguire il display del riquadro che circonda l’oggetto. Come verrà mostrato nel modello dinamico, il display delle informazioni avviene solo se la percentuale di riconoscimento è sopra un certo valore percentuale.

I vincoli sulle coordinate e dimensioni verranno determinati in fase implementativa.

## Object Diagram

### 

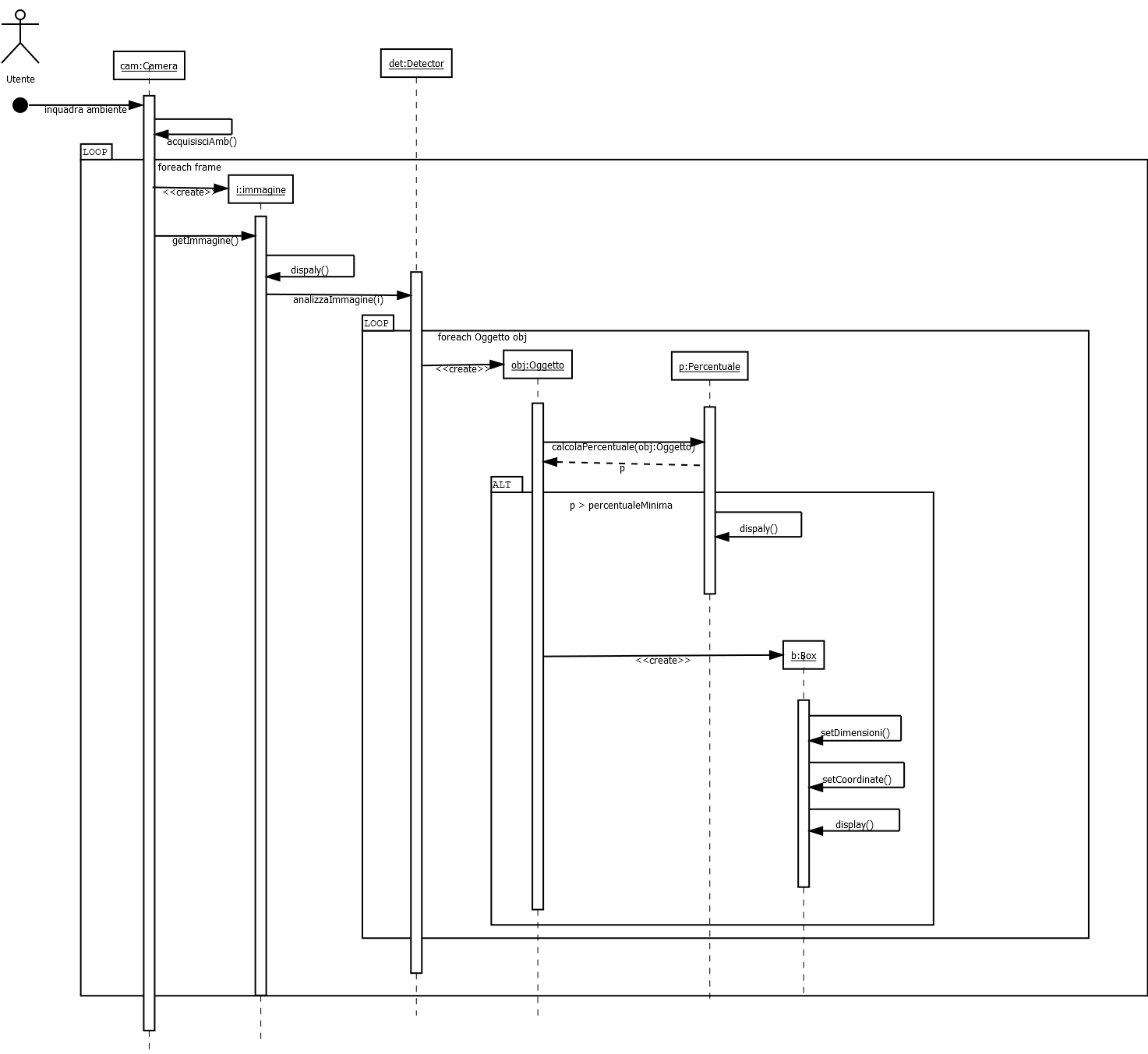
*Figura 6.2: Object Diagram*

Il diagramma in figura mostra una situazione tipica che può verificarsi durante l’uso dell’applicazione atta al riconoscimento di oggetti.

* La camera del dispositivo acquisice l’ambiente circostante. Ogni frame sarà un’immagine differente, supponiamo nell’esempio di avere una singola immagine
* L’immagine acquisita dalla fotocamera contiene 4 differenti oggetti.
* Ogni oggetto ha le proprie coordinate e le proprie dimensioni. Poiché i vincoli sulle stesse sono da determinarsi in fase implementativa vengono indicati nel diagramma con cX; cY; L; A.
* Ogni oggetto ha la propria percentuale con cui viene riconosciuto come l’oggetto specifico per cui è avvenuto l’addestramento.
* Se la percentuale è sopra un certo valore, che al momento supponiamo essere 0.80 cioè affidabilità all’80%, viene realizzata la box che deve circondare l’oggetto a schermo.
* Per gli oggetti 1 e 3 viene determinato il riquadro, avente le proprie coordinate e le proprie dimensioni. Il discorso relativo ad esse è analogo a quanto detto in merito agli Oggetti.
* Box e percentuali degli oggetti 1 e 3 verranno mostrate a schermo, quelle degli oggetti 2 e 4 no in quanto la percentuale è al di sotto dello 0.80.

# Dynamic Model

## Inquadra Ambiente



*Figura 7.1: Sequence diagram, inquadra ambiente*

L’utente inquadra l’ambiente circostante con la fotocamera del dispositivo che si occuperà dell’acquisizione dello stesso; quindi viene acquisito un frame di ciò che inquadra la fotocamera.

L’immagine viene mostrata a schermo ed analizzata dal Detector. Per ogni oggetto che viene riconosciuto viene calcolata la percentuale e se essa è al di sopra di un valore di default viene ne eseguito il display. Sempre se il valore è al di sopra di quello di default vengono determinate posizione e coordinate della box che viene mostrata a schermo.

Se il valore è invece inferiore a quello di default allora non viene eseguito il display di nulla.

Il tutto viene eseguito per ogni frame che viene acquisito dalla fotocamera.