


Fruit Ninja Webcam

Titolo del progetto: Fruit Ninja Webcam
Alunno/a: Curiale Alessandro
Classe: Info 3BB
Anno scolastico: 2022/2023
Docente responsabile: Geo Petrini

1	Introduzione	4
1.1	Informazioni sul progetto	4
1.2	Abstract	4
1.2.1	Risultati	4
1.3	Scopo	4
1.3.1	Scopi lavorativi	4
1.3.2	Scopi didattici	5
2	Analisi	6
2.1	Analisi del dominio	6
2.2	Analisi e specifica dei requisiti	6
2.3	Use case	8
2.4	Pianificazione	9
2.5	Pianificazione	1
2.6	Analisi dei mezzi	1
2.6.1	Software	1
2.6.2	Hardware	1
3	Progettazione	2
3.1	Design dell'architettura del sistema	2
3.1.1	Menu Principale	2
3.1.2	Impostazioni	2
3.1.3	Test della webcam	2
3.1.4	Gioco	2
4	Implementazione	3
4.1	Entity	3
	3
4.2	Sprite	3
	3
4.3	GamePanel	4
4.4	SettingPanel	6
4.5	HandGesture	6
4.6	Game Menu	7
4.7	Main	8
5	Test	9
5.1	Protocollo di test	9
5.2	Risultati test	10
5.3	Mancanze/limitazioni conosciute	11
5.3.1	Hand Tracking + taglio	11
5.3.2	Gestioni vite	11
5.3.3	Gestione morte	11
5.3.4	Gestione punti	11
5.3.5	Gestione tempo	11
5.3.6	Macchie + tagli sfondo	11
5.3.7	Sdoppiamento elemento	11
6	Consuntivo	12
7	Conclusioni	13
7.1	Sviluppi futuri	13
7.2	Considerazioni personali	13
8	Bibliografia	14
8.1	Sitografia	14
8.2	Indice delle figure	14
9	Allegati	14

	SAMT – Sezione Informatica	Pagina 3 di 23
	Esempio di documentazione	

1 Introduzione

1.1 Informazioni sul progetto

- Allievo: Curiale Alessandro
- Docente Responsabile: Geo Petrini
- Scuola: CPT Trevano
- Sezione Informatica
- Classe I3BB
- Inizio Progetto: 09.09.2022
- Fine Progetto: 23.12.2022

1.2 Abstract

Questa documentazione contiene tutto quello che sono riuscito a fare sul progetto assegnatomi, inoltre potrete vedere la pianificazione tramite gantt.

Questo progetto purtroppo non è stato completato ma potete fin dove sono arrivato e quali problemi ho riscontrato.

Questo progetto doveva essere un applicativo ricreativo per divertirsi ma alla fine non è altro un applicativo che permette all'utente di vedere le webcam collegate, vedere dei frutti saltare senza la possibilità di interagirvi.

Ho avuto parecchi problemi causati da litiengine, ho speso molte ore su questa libreria perché mi sembrava un'opportunità per imparare ad usare quella libreria:

- Ho perso molto tempo a documentarmi su questa libreria anche se possedevo una documentazione fatto male e a volte mancavano dei punti senza la quale ho dovuto improvvisare.
- La parte più rognosa è stata quella del tempo di compilazione, ogni volta che modificavo il programma e volevo testarlo, dovevo attendere minimo 4.31 min per poi attendere altri 2 min per l'avvio del programma.

Mi sono ritrovato a rifare il progetto con quattro settimane alla consegna dove dovevo rifare tutto tranne la gestione dei thread.

1.2.1 Risultati

Il risultato finale è deludente, speravo di riuscire a fare di più, ma sono consapevole che oltre ad essermi perso a causa di Litiengine in classe non ho lavorato pienamente sul mio progetto, ma vagavo per l'aula aiutando un po' i miei compagni.

Il Fruit Ninja sviluppato, una volta avviato il gioco, genera solamente i frutti in modo casuale, manca lo sviluppo dell'algoritmo per il rilevamento delle mani, sono riuscito a implementare un algoritmo per rilevare la pelle.


C'è la possibilità di visualizzare i dispositivi webcam collegati senza la possibilità di selezionarne uno effettivamente.

1.3 Scopo

Lo scopo principale è quello di riprodurre il gioco Fruit Ninja apportandone alcune modifiche, oltre a questo, essendo il primo progetto serve anche a imparare a saper gestire in maniera autonoma il proprio tempo, vedere effettivamente le proprie abilità e scoprire i propri limiti, infatti posso dire che questo progetto è stato troppo grande per me, si può notare dai risultati.

1.3.1 Scopi lavorativi

Ampliare le conoscenze del linguaggio java, riprodurre il gioco *Fruit Ninja* ma con alcune modifiche, ad esempio quello di usare un hand tracking invece del classico touch.

	SAMT – Sezione Informatica	Pagina 5 di 23
	Esempio di documentazione	

1.3.2 Scopi didattici

Imparare ad implementare delle interfacce che sfruttano dei dispositivi esterni, imparare a gestire un progetto in modo autonomo, conoscere le attuali potenzialità che possiedo e posso dire con certezza che la pianificazione creata non è stata seguita molto, questo mi fa capire che la gestione iniziale è molto importante come tenere una documentazione sul lavoro svolto, tramite diari o altro.

2 Analisi

2.1 Analisi del dominio

Questo progetto consiste nel riprodurre il gioco Fruit Ninja apportando alcune modifiche, il gioco finitò dovrà funzionare su computer aventi come sistema operativo Windows 10/11 con almeno una videocamera collegata. Questo gioco non ha fasce di età, chiunque vuole tenersi in movimento con questo gioco è ben accetto.

2.2 Analisi e specifica dei requisiti

ID: REQ-001	
Nome	Rilevamento delle mani
Priorità	1
Versione	1.0
Note	
Sotto requisiti	
001	Rilevamento dei movimenti
002	Webcam connessa al pc
003	Rilevamento massimo di 4 alla volta e consigliato 1,5m di stanza per un ambiente di gioco ottimale

ID: REQ-002	
Nome	Generazione frutti
Priorità	1
Versione	1.0
Note	I frutti possono essere di 5 tipi (anguria, cocco, ciliegia, arancia, banana)
Sotto requisiti	
001	Tipo di frutto random
002	Il numero dei frutti che appaiono è costante
003	

ID: REQ-003	
Nome	Tagliare i frutti
Priorità	1
Versione	1.0
Note	
Sotto requisiti	
001	Movimento deciso della mano
002	In concomitanza al rilevamento massimo di 4 mani, il numero massimo di tagli che si possono effettuare alla volta sono 4
003	Ogni frutto tagliato assegna 2pt all'utente
004	I frutti tagliati creano delle macchie di succo
005	Il taglio fa rimanere il segno sullo sfondo del gioco per un po'

ID: REQ-004	
Nome	Gestione menu
Priorità	1
Versione	1.0
Note	
Sotto requisiti	
001	Compare all'avvio
002	Compare quando si perde
003	

ID: REQ-005	
Nome	Gestione Game Over
Priorità	1
Versione	1.0
Note	
Sotto requisiti	
001	Controllare se si hanno mancati 3 frutti
002	Controllare se ha tagliato una bomba

ID: REQ-006	
Nome	Gestione zona ti taglio
Priorità	1
Versione	1.0
Note	
Sotto requisiti	
001	Mantiene delle proporzioni fisse cioè 1920 x 1080
002	Overflow con decorazioni
003	

2.3 Use case

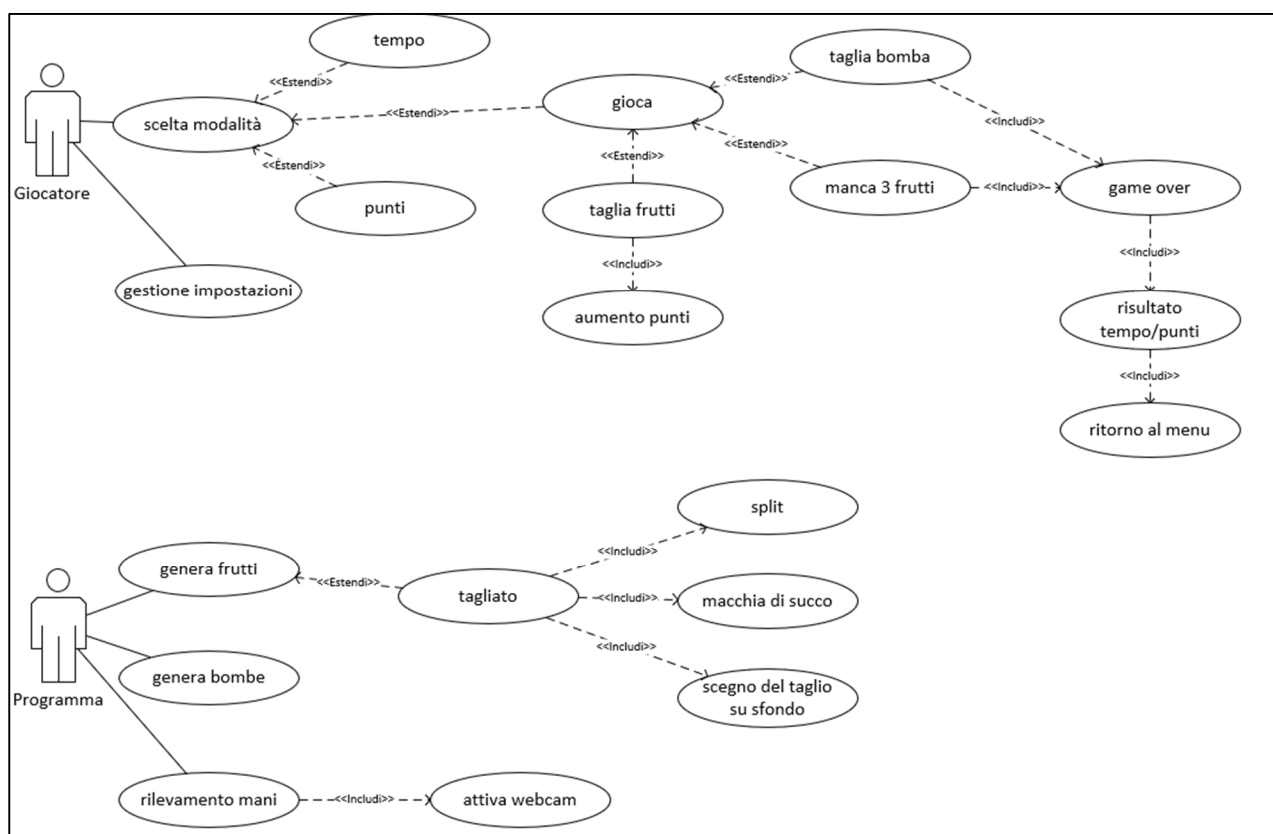


Figura 1 UML USE CASE Fruit Ninja

2.4
Pianificazione

Qui potete vedere come ho pianificato le varie task, le loro durate e le varie precedenze.

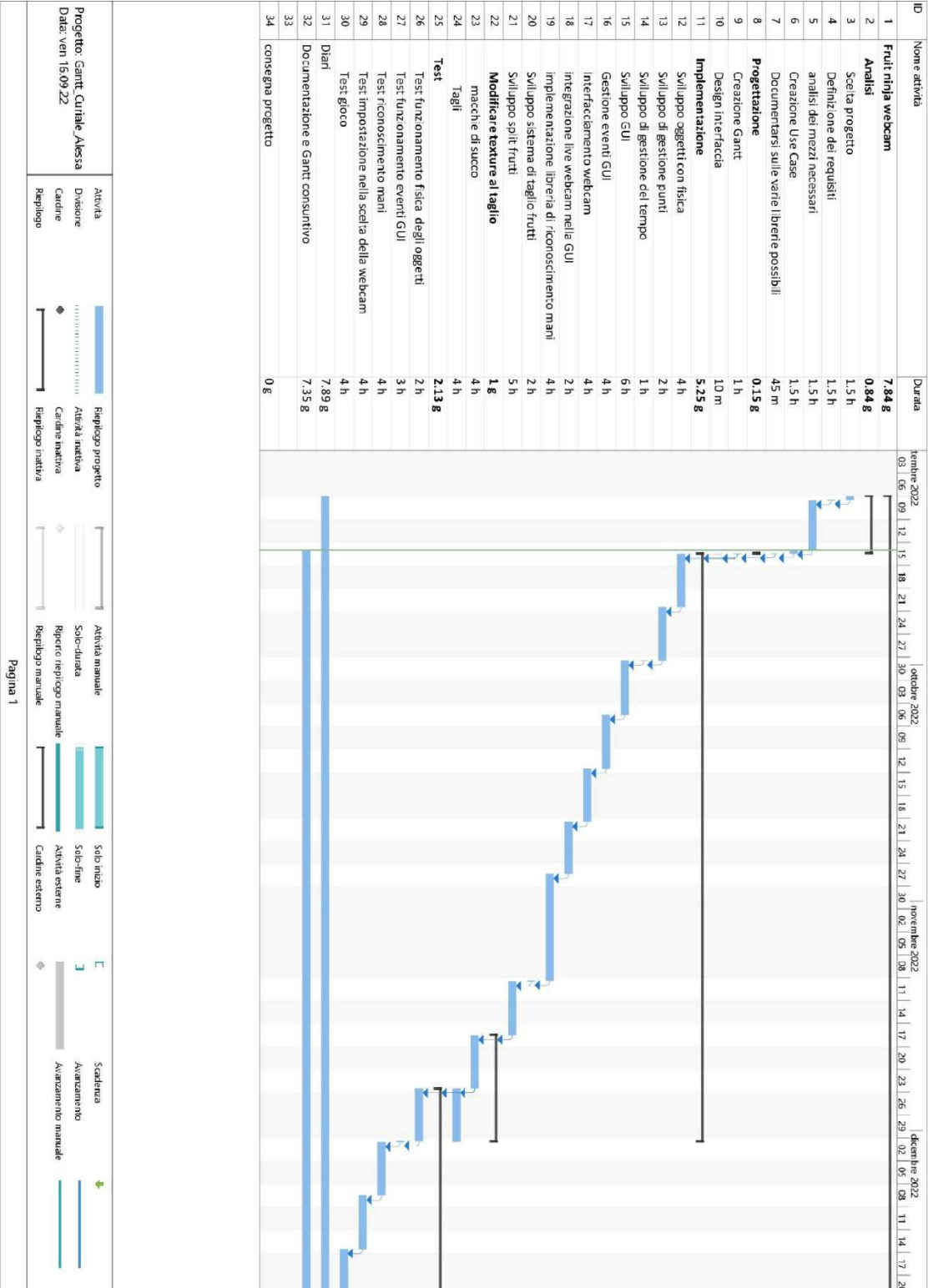


Figura 2
Gantt Iniziale

2.5 Pianificazione

Gantt + spiegazione

2.6 Analisi dei mezzi

Per realizzare questo progetto è stato utilizzato un PC scolastico, alcune webcam fornite dalla scuola ed il software NetBeans.

2.6.1 Software

Software utilizzati per la realizzazione di questo progetto:

- NetBeans 13
- Microsoft Word
- Microsoft Visio
- Microsoft Project
- GitHub con Git

Librerie usate nel programma:

- OpenCV
- JavaCV
- OpenBlas
- Sarxos

2.6.2 Hardware

Il progetto per funzionare correttamente deve essere eseguito su una macchina windows 10/11 con queste specifiche hardware minime:

- Intel(R) Core(TM) i5 CPU
- 8.0 GB
- Scheda video integrata

Caratteristiche pc usato:

- Intel(R) Core(TM) i7-9700 CPU
- 32.0 GB
- NVIDIA GeForce RTX 2060

Webcam utilizzata:

- Logitech C922 Pro Stream Webcam
- Logitech C920 Pro HD Webcam

Qualità webcam consigliata:

- 720p

Titolo del progetto:	Fruit Ninja Webcam
Alunno/a:	Curiale Alessandro
Classe:	Info 3BB
Anno scolastico:	2022/2023
Docente responsabile:	Geo Petrini

3 Progettazione

Questo capitolo descrive esaurientemente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all'esecutore di evitare fraintendimenti e imprecisioni nell'implementazione del prodotto.

3.1 Design dell'architettura del sistema

Il gioco possiede più interfacce:

- Menu principale
- Impostazioni
- Test della webcam
- Il gioco

3.1.1 Menu Principale

Il menu principale appare all'avvio dell'applicazione e contiene semplicemente 4 pulsanti

- Gioca a tempo
- Gioca a punti
- Impostazioni
- Webcam test

Alla premuta del pulsante scelto l'interfaccia cambia di conseguenza.

3.1.2 Impostazioni

Genera una lista di radio button in base ai dispositivi webcam trovati, selezionando la webcam e uscendo si sarà impostata la webcam.

3.1.3 Test della webcam

Il test della webcam serve all'utente se desidera verificare se la webcam in utilizzo funzioni correttamente; infatti, il gioco accenderà la webcam mostrando così quello che vede.

3.1.4 Gioco

Nella schermata di gioco sarà visualizzato uno sfondo che richiama un tagliere di legno, le vite dell'utente, il suo punteggio o il suo tempo, il tempo e il punteggio verrà visto in base alla modalità scelta dall'utente.

4 Implementazione

4.1 Entity

```
public class Entity {
    public double x, y, a, b, c, incX;
    public double rotation;
    public double speed;
    public boolean visible = false;
    public boolean cut = false;
    public BufferedImage image;
}
```

Figura 3 Classe entity

Questa classe serve per istanziare un'entità del gioco, esso può essere un frutto o una bomba.

4.2 Sprite

```
public Sprite(GamePanel gp, int size, String imagePath, double a, double b, double c) {
    this.gp = gp;
    this.size = size;
    this.a = a;
    this.b = b;
    this.c = c;
    Random ranInc = new Random();
    if(ranInc.nextInt(0, 2) == 1){
        this.x = 1920;
        this.incX = -0.1;
    }else{
        this.x = 0;
        this.incX = 0.1;
    }
    this.x = 1920;
    if (imagePath != null) {
        try {
            setImage(imagePath);
        } catch (IOException ex) {
            Logger.getLogger(Sprite.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

Figura 4 Costruttore della classe Sprite

Questa classe estende Entity, infatti sfrutta i suoi attributi e li setta di conseguenza.

```
public void update() {
    this.x += this.incX;

    this.y = a * Math.pow(x/ampiezza-spostX, 2)+ b * x + c;

    if(!isVisible()){
    }
}
```

Figura 5 Metodo che aggiorna la posizione del oggetto

Metodo che mi aggiorna la posizione del frutto in base alla sua funzione e alla sua x attuale.

```
public void draw(Graphics2D g2) throws IOException {
    if (image != null) {
        BufferedImage imageToDraw = image;

        g2.drawImage(imageToDraw, (int)x, (int)y, size, size, null);
    }
}
```

Figura 6 Metodo che mi disegna l'oggetto

Metodo che mi disegna l'entità sul panel.

4.3 GamePanel

```
public void startGameThread(){
    punteggio.setText("0");
    this.add(punteggio);

    initComponents();

    gameThread = new Thread(this);
    gameThread.start();
}
```

Figura 7 Metodo che mi avvia il gioco

Una volta avviato il gioco, viene creato un Thread e viene avviato.

Questo Thread si occuperà di generare i frutti ogni tot e di stampare un log a terminale.

```
public void run() {

    int secondi = (int) (System.currentTimeMillis() / 1000);
    int previuTime = (int) (System.currentTimeMillis() / 1000);
    double drawIntervall = 1000000000 / FPS;
    double deltaIntervall = 0;
    long previousTime = System.nanoTime();
    long currentTime;
    double previousSecond = 0;
    while (gameThread != null) {
        currentTime = System.nanoTime();
        deltaIntervall += (currentTime - previousTime) / drawIntervall;

        LOGGER.info("Gioco in esecuzione\n");
        if (deltaIntervall >= 1) {
            update();
            repaint();
            deltaIntervall--;
        }
        jLabel1.setText(secondi + "");
        secondi = (int) (System.currentTimeMillis() / 1000 - previuTime);

        if (secondi % 2 == 1 && secondi != 0 && secondi != previousSecond) {
            previousSecond = secondi;
            Random ran = new Random();
            //a[0.08; 0.5] b[0.0; 1.0] c[-20;10]
            double aRandom = ran.nextDouble() * 2 + 0.08;
            double bRandom = ran.nextDouble() * 4 - 3;
            double cRandom = ran.nextDouble() * 10 + 520;
            int fruitRandom = ran.nextInt(0, 5);
            String stringPath = "frutto"+ fruitRandom + ".png";
            Sprite m = new Sprite(this, 120, stringPath, aRandom, bRandom, cRandom);

            entities.add(m);
        }
    }
}
```

Figura 8 Thread che mi genera i frutti

4.4 SettingPanel

Classe si occupa di generare dei radio button in base ai dispositivi webcam che rileva

```
public void createList() {

    int count = (int)videoInput.listDevices();

    String[] descriptions = new String[count];

    ButtonGroup group = new ButtonGroup();
    JRadioButton btn = null;

    for (int i = 0; i < descriptions.length; i++) {
        descriptions[i] = videoInput.getDeviceName(i).getString();
    }
    for (int i = 0; i < descriptions.length; i++) {
        System.out.println("");
        if(i == 0){
            btn = new JRadioButton(i + " " + descriptions[i]);btn.setSelected(true);
        }else{
            btn = new JRadioButton(i + " " + descriptions[i]);
        }
        group.add(btn);
        this.add(btn);
    }
    initComponents();
}
```

Figura 9 Metodo che rileva le webcam

4.5 HandGesture

Classe che dovrebbe occuparsi nel rilevare i gesti delle mani, purtroppo non è completata, infatti rileva solo la pelle tramite il colore, infatti una persona di colore non verrebbe rilevata.

4.6 Game Menu

Classe panel che racchiude 4 pulsanti che azionano i vari panel.

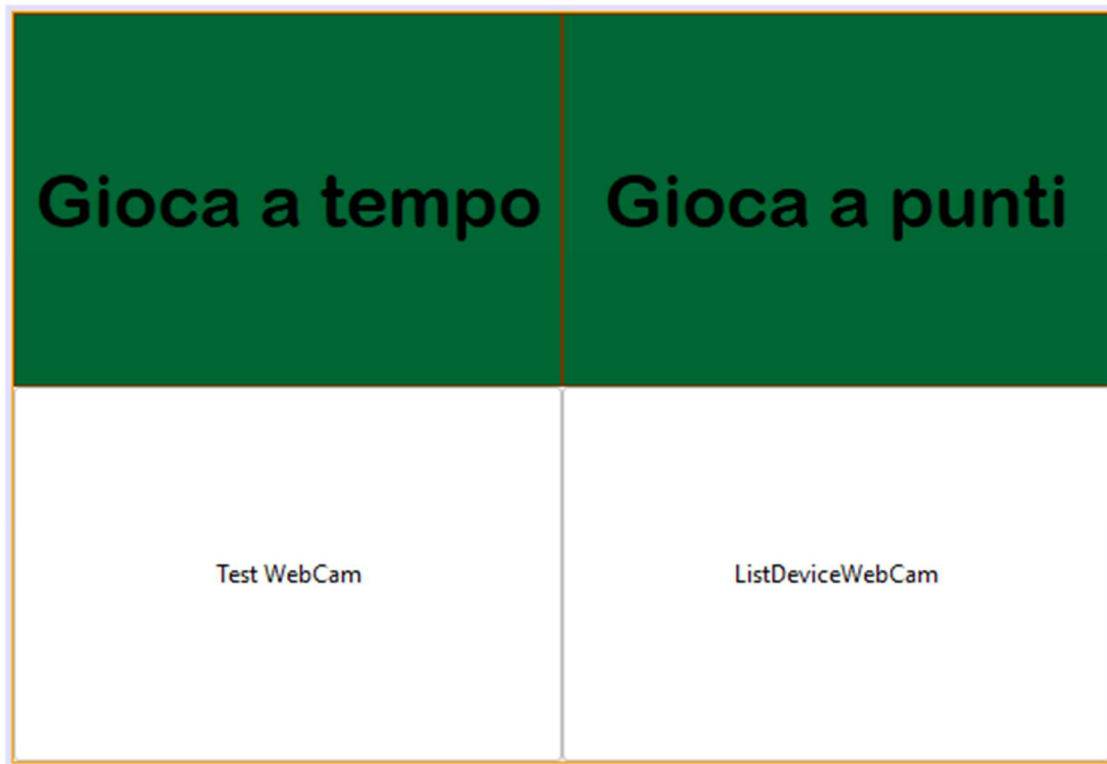


Figura 10 Gui

4.7 Main

```

public void startGame(){
    this.gamePanel.startGameThread();
}

public void startCam(){
    handGesture.startCapture();
}

public void createList(){
    this.settingsPanel.createList();
}

/** This method is called from within the constructor to initialize the form ...5 lines */
@SuppressWarnings("unchecked")
Generated Code

private void gameMenuPropertyChange(java.beans.PropertyChangeEvent evt) {
    // TODO add your handling code here:
    if (evt.getPropertyName() == "playI"){
        this.gameMenu.setVisible(false);
        this.gamePanel.setInPoints(false);
        this.startGame();
        startCam();
    }

    if (evt.getPropertyName() == "playP"){
        this.gameMenu.setVisible(false);
        this.gamePanel.setInPoints(true);
        this.startGame();
        startCam();
    }

    if (evt.getPropertyName() == "capture"){
        this.gameMenu.setVisible(false);

        this.startCam();
    }

    if (evt.getPropertyName() == "list"){
        this.gameMenu.setVisible(false);
        this.createList();
    }
}

```

Figura 11 Classe Main che verifica il pulsante premuto

Classe principale che contiene tutti i panel non visibili tranne il Game Menu, infatti al click del pulsante aziona un evento che a sua volta mostra il panel desiderato avviando l'opzione decisa.

5 Test

5.1 Protocollo di test

Di seguito potrete vedere i test per verificarne il corretto funzionamento

Test Case:	TC-001	Nome:	Test rilevamento mani
Riferimento:	REQ-001		
Descrizione:	Rilevamento mani all'avvio dell'applicazione		
Prerequisiti:	Webcam impostato correttamente		
Procedura:	1. Prova rilevamento Massimo 4 mani 2. Prova 1metro di distanza per trovare la distanza ottimale 3. Prova rotazione delle mani		
Risultati attesi:	-Rileva un massimo 4 mani -1 metro distanza abbastanza ottimale -la rotazione delle mani non influisce sul rilevamento		

Test Case:	TC-002	Nome:	Test Generazione frutti
Riferimento:	REQ-002		
Descrizione:	Generazione casuale di frutti		
Prerequisiti:			
Procedura:	1. Iniziare una partita e verificare che I frutti vengono generate in modo casuale		
Risultati attesi:	- allo start della partita I frutti vengono generate casualmente e in modo costante		

Test Case:	TC-003	Nome:	Test taglio
Riferimento:	REQ-003		
Descrizione:	- Test per controllare se il taglio viene effettuato correttamente - Max 4 tagli in contemporanea		
Prerequisiti:	- TC-001 funzionante		
Procedura:	- Iniziare una partita e verificare il segno di taglio che seguela mano ad uno scatto - Se effettuato su un frutto I punti aumentano di 2, il taglio viene segnato sul background e anche la macchia di succo		
Risultati attesi:	- - allo start della partita si vede un segno(taglio) che segue la mano e taglia il frutto aumentado i punti e mostrando le macchie di succo e I tagli sul background - -si vedranno un Massimo di 4 tagli alla volta		

Test Case:	TC-004	Nome:	Test Gestione menu
Riferimento:	REQ-004		
Descrizione:	Gestione generale del menu principale		
Prerequisiti:	-		
Procedura:	Avviare l'applicazione e verificare il menu principale e verificare che le varie opzioni siano funzionanti. Inoltre la ricomparsa del menu nel caso si perde la partita.		
Risultati attesi:	<ul style="list-style-type: none"> - menu principale subito visibile con tutte le opzioni funzionanti - quando si perde una partita ricompare un menu 		

Test Case:	TC-005	Nome:	Test Gestione gameover
Riferimento:	REQ-005		
Descrizione:	Gestione dell'azione che verifica il game over		
Prerequisiti:	-TC-003		
Procedura:	Avviare una partita e verificare se le vite diminuiscono nel caso in cui si mancano i frutti o vengono tagliate delle bombe		
Risultati attesi:	- mancare 3 frutti o colpire una bomba finisce la partita		

Test Case:	TC-006	Nome:	Test dimensioni applicazioni
Riferimento:	REQ-006		
Descrizione:	Gestione generale delle dimensioni dell'applicazione		
Prerequisiti:	-		
Procedura:	Avviare l'applicazione e modificare la dimensione della finestra.		
Risultati attesi:	- al cambio di dimensione prestabilita appare un overflow sui lati in eccesso		

5.2 Risultati test

ID	Risultato	Note	Data
TC-001	Non passato	<u>Il gioco rileva tutta la palla, questo implica che rileva anche altro oltre alle mani.</u>	23.12.2022
TC-002	Passato	<u>I Frutti vengono generati casualmente correttamente</u>	9.12.2022
TC-003	Non passato	<u>I Tagli non vengono effettuati a causa della mancanza dell'algoritmo</u>	23.12.2022
TC-004	Non passato	<u>Menu iniziale funzionante, ma non si può perdere, quindi non riappare il menù.</u>	23.12.2022
TC-005	Non passato	<u>Non vengono gestite le vite dell'utente, infatti parte da 3 vite ma ne può perdere all'infinito.</u>	23.12.2022
TC-006	Non passato	<u>L'applicazione ha dimensioni fissi, l'utente non è possibilitato a modificarne le dimensioni, quindi non appare nessun overflow</u>	23.12.2022

5.3 Mancanze/limitazioni conosciute

5.3.1 Hand Tracking + taglio

Non sono riuscito a completare l'algoritmo per il riconoscimento delle mani, sono riuscito solo a rilevare la pelle, di conseguenza non ho potuto creare l'evento del taglio.

5.3.2 Gestioni vite

Ogni volta che un frutto cade si perde una vita, ma genera un'eccezione a causa del tentativo di eliminare l'oggetto, inoltre perdo vite infinite

5.3.3 Gestione morte

Non ho avuto tempo per gestire la morte tramite le vite perse.

5.3.4 Gestione punti

Non sono riuscito ad usare un label nel panel per tenere i punti, al taglio dei frutti aumento i punti senza la possibilità di verificarne la correttezza.

5.3.5 Gestione tempo

Come con la gestione dei punti ho avuto dei problemi ad usare i label nel panel, a passare del tempo non aggiornò il label, non è stato aggiunto al panel.

5.3.6 Macchie + tagli sfondo

Sempre collegato al rilevamento delle mani, e all'evento dei tagli non sono riuscito a generare degli effetti per lasciare dei tagli o delle macchie di succo sullo sfondo

5.3.7 Sdoppiamento elemento

Collegato all'evento del taglio non gestito, non ho potuto sdoppiare il frutto tagliato.

6 Consuntivo

Di seguito c'è il Gantt consuntivo di come ho svolto le varie task, posso dire che tutt'altro di come mi aspettavo, speravo di metterci molto meno in alcune task, invece si sono rilevate molto più lunghe del previsto.

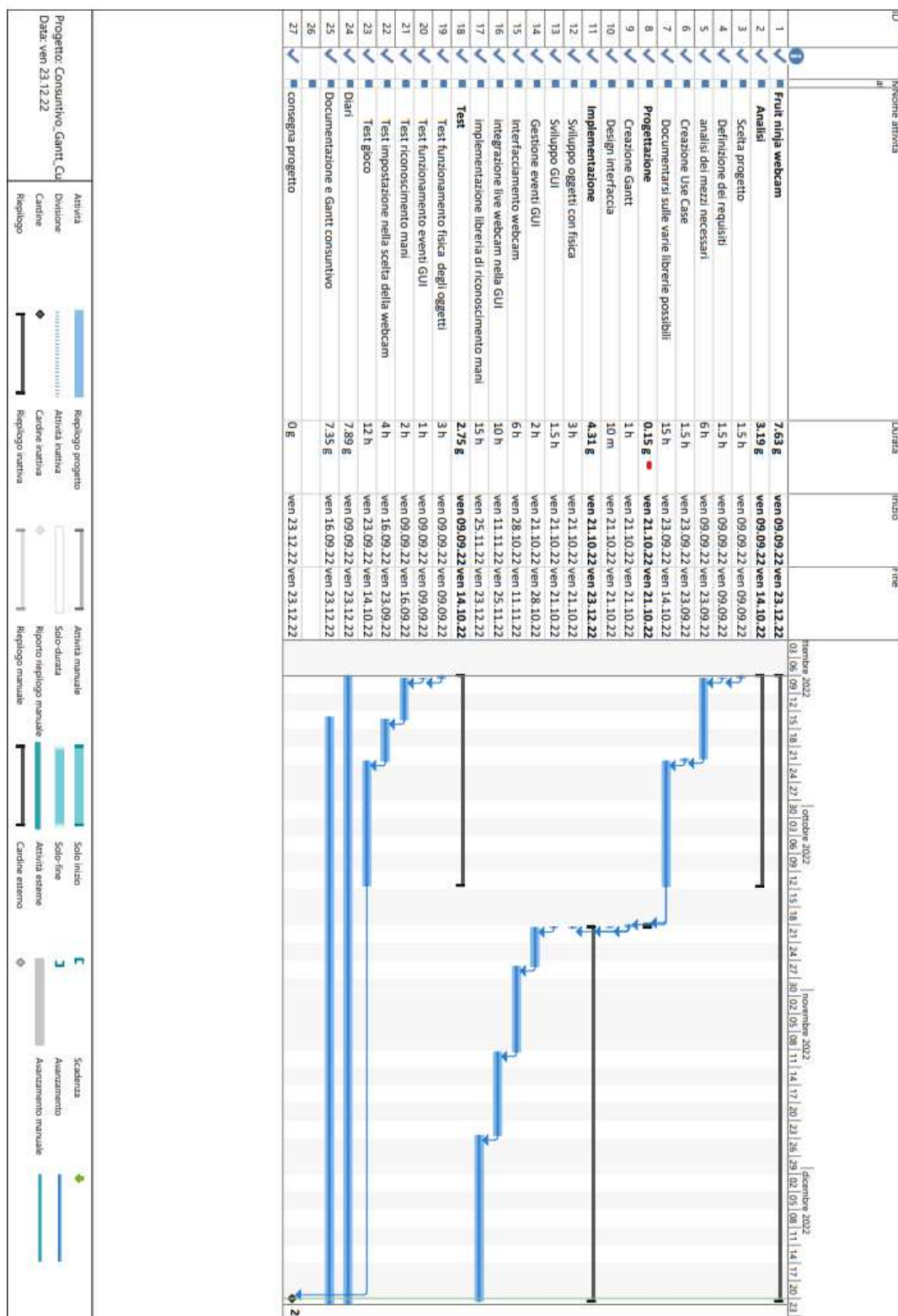


Figura 12 Gantt consuntivo

7 Conclusioni

In conclusione posso dire che il mio progetto sarebbe stato solo a scopo ludico, purtroppo non sono riuscito a portarlo a termine, spero che in un futuro io lo porti a termine.

7.1 Sviluppi futuri

Il prodotto come prima cosa deve essere finito, in un futuro però si potrebbe aggiungere:

- Scelta scia del taglio
- Aggiunti frutti che se tagliati danno de power up
- Scelta scenario di gioco (cambio sfondo)
- Classifica con i migliori punteggi delle varie modalità
- Aggiunto il multiplayer con la possibilità di sfidare gli amici
- Aggiunta effetti sonori
- Possibilità di ridimensionare la finestra
- Aggiungere la possibilità di mettere in pausa

7.2 Considerazioni personali

Posso dire che non sono soddisfatto del risultato finale ma ho certamente capito dove ho sbagliato, infatti come prima cosa avrei dovuto essere meno ambizioso ma più realista, infatti il progetto è troppo grande per le mie abilità.

In considerazione tengo conto anche al fatto che ho perso tempo nell'aiutare i compagni, riducendo così il tempo dedicato al mio progetto.

Questo progetto comunque mi ha aiutato molto a maturare e a gestire meglio il tempo a disposizione come la conoscenza delle mie abilità, inoltre ho imparato che esistono non solo librerie utili e fatte bene ma come Litiengine che è fatta male ed è poco documentata.

Devo dire in oltre che questo progetto magari con qualche mano in più sarebbe stato bello da portare a termine.

Infine ringrazio Petrini, il mio docente responsabile che mi ha aiutato e ha cercato anche lui di documentarsi su Litiengine.

8 Bibliografia

8.1 Sitografia

- <https://github.com/amulyaarunb/Hand-gesture-recognition/blob/master/HandGesture.java>, 23.12.2022
- <https://www.youtube.com/watch?v=Y6oLbRKwmPk>, 23.12.2022
- <https://stackoverflow.com/questions/18941840/add-buttongroup-to-jpanel>, 9.12.2022
- <https://litiengine.com/>, 2.12.2022

8.2 Indice delle figure

Figura 1 UML USE CASE Fruit Ninja	8
Figura 2 Gantt Iniziale.....	9
Figura 3 Classe entity	3
Figura 4 Costruttore della classe Sprite	3
Figura 5 Metodo che aggiorna la posizione del oggetto.....	4
Figura 6 Metodo che mi disegna l'oggetto.....	4
Figura 7 Metodo che mi avvia il gioco	4
Figura 8 Thread che mi genera i frutti	5
Figura 9 Metodo che rileva le webcam.....	6
Figura 10 Gui	7
Figura 11 Classe Main che verifica il pulsante premuto	8
Figura 12 Gantt consuntivo	12

9 Allegati

Elenco degli allegati, esempio:

- USE CASE
- Gantt iniziale e consuntivo
- Diagramma delle classi
- Immagini usate
- Mandato e/o QdC
- Prodotto finale