

# Progetto RTOS

Berselli Werther  
Di Blasi Fabrizio

## Titolo:

Image processing. Simulate a fixed camera looking at a given portion of an animated screen and write 4 periodic tasks performing different computations on the same image and displaying the results on other images. For example, task1 displays the histogram, task2 displays the image after applying a threshold, task3 displays the difference between consecutive frames, task4 displays the result of a filter. The user must be able to vary some parameters of such tasks.

## Guida per l'utente:

Il programma ha il compito di analizzare uno stream video partendo da un riquadro tracciato a piacere dall'utente.

La schermata iniziale è la seguente:



All'utente sarà permesso di scegliere alcuni iper parametri, che verranno utilizzati per la computazione, ne sono 2 **FPS** e **Bit for channel**.

Con **FPS** si indicano i frame al secondo di cattura della telecamera, mentre con **Bit for channel** si impostano quanti bit si vogliono per ogni canale RGB.

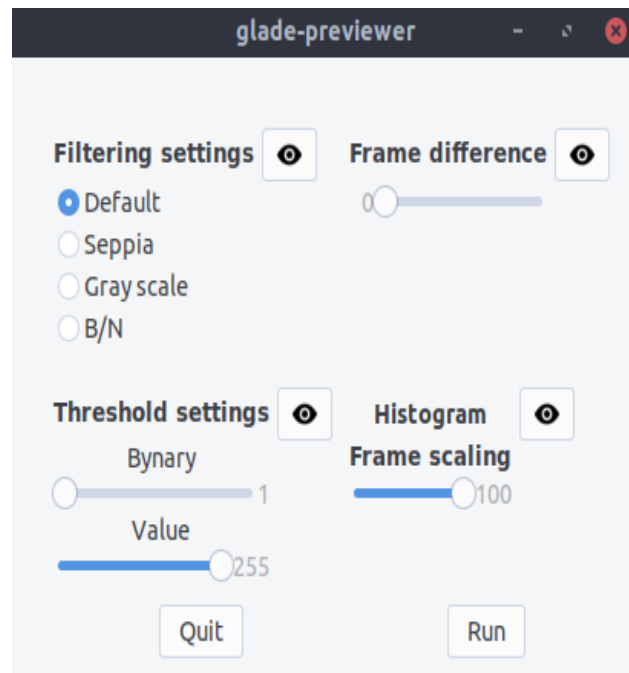
All'inizio sarà "clickabile" solamente il tasto "**Select Area**" che permetterà all'utente di selezionare la parte di schermo che si vuole processare.

Una volta selezionata la porzione di schermo, si potrà premere il tasto "**Confirm**" il quale farà comparire a video le 5 aree di lavoro (videocamera, e 4 thread).

In questo modo si dà un'idea dell'ingombro sullo schermo dei 4 riquadri.

Infine, col tasto “**Cancel**” si ritorna alla scelta degli iper parametri, altrimenti si avvia la computazione tramite il tasto “**Start**”.

Una volta premuto **Start**, verrà visualizzato il seguente frame:




questo frame permette all’utente di impostare dei parametri a piacere.

Questo frame rimarrà sempre visibile e permetterà all’utente di modificare alcuni parametri che modificheranno in tempo reale l’output dei thread di analisi.

I parametri da poter impostare sono 4:

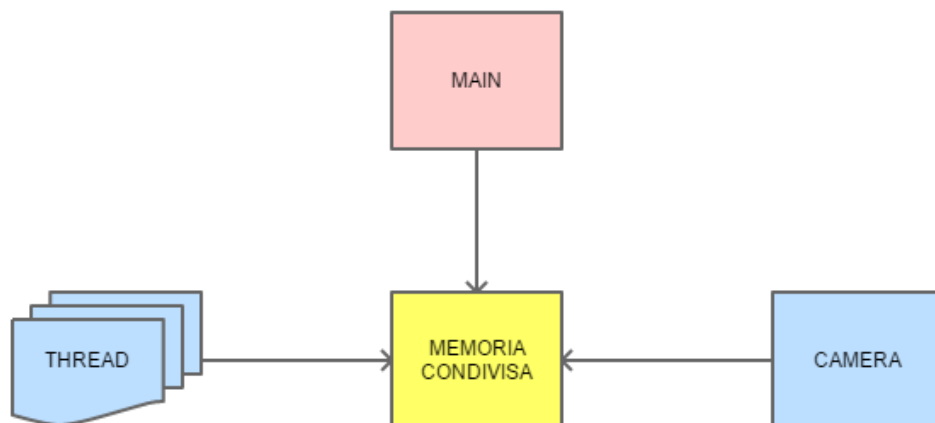
- Filter Settings:
  - default : riproduce l’immagine senza modificarne il colore
  - seppia : riproduce l’immagine applicando un filtro seppia
  - gray scale : modifica i colori dell’immagine catturata in una scala di grigi
  - B/N : riproduce l’immagine in bianco e nero
- Frame Difference
  - tramite lo “*slider*” è possibile scegliere l’intervallo per il calcolo della differenza pixel-pixel tra due fotogrammi prelevati in istanti diversi. Inizialmente si preleva quello ad istante 0, successivamente, sulla base della scelta dell’utente si aspetteranno “n” fotogrammi per prelevare il secondo e poter infine calcolare la differenza.
- Frame Scaling

- L'utente tramite lo *slider* potrà ridimensionare l'immagine visualizzata nella finestrella. In questo modo si dà la facoltà di analizzare un grande riquadro, e far rientrare le 4 caselle di analisi all'interno dello schermo
- Frame threshold
  - Tramite il doppio *slider* l'utente potrà selezionare la tipologia di soglia
    - binary
    - binary inverted
    - threshold truncated
    - threshold to zero
    - threshold to zero inverted
  - e potrà scegliere la soglia tra 0 – 255 del valore della scala RGB

Affianco alla Label delle impostazioni dei thread è situata un'iconcina  che è cliccabile. Tramite questa icona è possibile chiudere la visualizzazione dei frame GUI di analisi.

È previsto un tasto di “quit” che fa arrestare tutta la computazione generale, senza dover ricorrere all'arresto da terminale

## Struttura del codice ed organizzazione della memoria condivisa:



## Organizzazione della memoria condivisa:

La gestione della memoria condivisa è svolta in questo modo:

Premuto il tasto Run la camera, scatta un'istantanea del riquadro scelto dall'utente.

Tale fotogramma sarà inserito in una struct gestita da semafori e da una variabile che indica lo stato di esecuzione.

Il semaforo della telecamera è impostato ad 1, quelli dei thread di analisi a 0, così facendo il task della telecamera sarà il primo a partire.

Tramite la variabile di stato che può assumere *"idle"* oppure *"analysis"* si può evitare che la telecamera vada a cambiare il fotogramma della memoria condivisa nel mentre che i thread di analisi lo stanno ancora analizzando.

La telecamera verrà *"sbloccata"* solamente quando tutti i thread di analisi avranno mandato a video la loro computazione, andrà quindi a prelevare un nuovo fotogramma senza che altri interferiscano, ed una volta terminato ricomincia il ciclo appena descritto.

## Note di progetto:

Per semplicità ed avere un'organizzazione più chiara il progetto è stato organizzato in più file cpp:

- Main del programma: manda a video i frame GUI
  - main.cpp
- Frame GUI con la quale interagisce l'utente
  - gui.cpp
- Il file cpp nel quale vengono definite le operazioni svolte dai thread di analisi è:
  - image\_processing.cpp

Tale file è organizzato in sezioni:

1. Funzioni che servono ad interagire con l'utente ed intercettare le sue scelte (in questa sezione è definita la funzione che traccia il rettangolo a schermo)
2. Le funzioni per la gestione dei thread di analisi, quindi la memoria condivisa
3. La dichiarazione delle funzioni per l'elaborazione del frame video catturato
4. La gestione dell'output a video, che anch'essa è gestita in mutua esclusione

e le seguenti librerie:

- Librerie usate per la gestione della GUI
  - image\_processing\_gui.cpp
- Librerie per l'analisi dei fotogrammi e gestione dei thread
  - image\_processing.h