

# RELAZIONE DI FINE TIROCINIO CURRICULARE

svolto dallo studente:

Leonardo Straccali 0000914784

iscritto al Corso di Studio in

L - ( 9254 ) INGEGNERIA INFORMATICA

presso:

ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

sul seguente argomento:

Sviluppo di un software per la costruzione di operatori di gruppo equivarianti non espansivi (GENEOs)

Studente:

*Straccali Leonardo*

(firma)

(firma per approvazione della relazione finale)

**Referente Struttura Ospitante:**

(inserire nome, cognome e ruolo all'interno alla struttura)

Patrizio Frosini, professore associato

*Patrizio Frosini*

(firma per approvazione della relazione finale)

Anno accademico 2022/2023

# SOMMARIO

<b>INTRODUZIONE</b>	<b>3</b>
<b>TECNOLOGIE USATE</b>	<b>4</b>
<b>SVILUPPO DEL PROGETTO</b>	<b>8</b>
<b>ALGORITMI IMPLEMENTATI</b>	<b>12</b>
<b>UTILIZZO DEL PROGRAMMA</b>	<b>18</b>
<b>POSSIBILI SVILUPPI FUTURI</b>	<b>22</b>
<b>CONCLUSIONE</b>	<b>23</b>
<b>SITOGRAFIA</b>	<b>24</b>
<b>RIFERIMENTI</b>	<b>25</b>



# INTRODUZIONE

L'obiettivo del tirocinio è stato lo sviluppo di un software per la costruzione di operatori di gruppo equivarianti non espansivi (GENEO) mediante l'inserimento in ingresso di permutanti e funzioni simmetriche. Gli operatori sopra citati, hanno una valenza nel campo dell'intelligenza artificiale, in quanto, possono essere applicati per costruire un nuovo paradigma di reti di apprendimento, basate sullo studio formale dello spazio topologico di questi operatori.

La prima fase del tirocinio si è concentrata sullo studio della ricerca del Professor Frosini, mediante gli articoli [\[1\]](#) e [\[2\]](#). Una volta presa conoscenza dell'argomento, è iniziata la fase di analisi, in cui è stato deciso che il software doveva avere uno scopo educativo, per facilitare gli studenti ad affacciarsi in modo visuale alla ricerca del professore. Dopo aver elencato i requisiti utente, di sistema e gli scenari dell'applicazione, sulla base essi ho scelto le specifiche del software, prediligendo tecnologie innovative, stabili e con una comunità attiva, in modo da ottenere un buon compromesso tra innovazione, usabilità e sicurezza.

Il risultato del tirocinio è un programma leggero, efficiente e compatibile con tutte le maggiori piattaforme, in grado di elaborare immagini date in ingresso attraverso funzioni simmetriche e restituire un GENEIO, normalizzato opportunamente per essere mostrato anch'esso come immagine.

Il software, essendo pensato per un utilizzo didattico, contiene molte descrizioni informative, in modo che uno studente che si affacci per la prima volta alla ricerca del professore, possa sperimentare gli strumenti matematici descritti dagli articoli con una metodologia semplice e visuale.



# TECNOLOGIE USATE

In questa sezione è presente l'elenco delle tecnologie usate nello sviluppo del progetto. La motivazione della scelta di tali tecnologie e la loro applicazione verrà discussa nel capitolo successivo.

**WebView:** è una tecnologia che consente agli sviluppatori di creare applicazioni desktop in grado di visualizzare contenuti web. In genere viene utilizzata per creare applicazioni ibride. È uno strumento che consente agli sviluppatori di creare applicazioni che possono essere eseguite su una varietà di piattaforme e dispositivi, sfruttando la potenza delle tecnologie Web, pur garantendo le prestazioni di un'applicazione nativa.

**Tauri:** è un framework per la realizzazione di applicazioni desktop leggere, sicure e veloci che utilizza la tecnologia WebView. È costruito sulla base del linguaggio di programmazione Rust, che ha come caratteristica principale l'attenzione alla sicurezza. Le applicazioni Tauri sono progettate per essere intrinsecamente sicure, con una serie di misure di sicurezza che impediscono l'esecuzione di codice dannoso. Alcune di queste misure sono: la limitazione dell'accesso ad alcune funzioni native, la fornitura di un canale di comunicazione sicuro tra i livelli nativo e web dell'applicazione e il sandboxing del widget webview. Nello specifico il sandboxing è una tecnica di sicurezza che consiste nell'isolare il widget da altre parti dell'applicazione, in questo modo, il widget può essere eseguito in un ambiente protetto che ne limita le azioni. Applicando questa strategia il widget può essere utilizzato per visualizzare pagine web esterne, garantendo che esso non abbia accesso alle risorse o alle funzionalità dell'applicazione. Il nome “sandboxing” deriva dal recinto riempito di



sabbia, comune nei parchi giochi, dove i bambini possono giocare e sperimentare in modo sicuro.

[Vue](#): è un framework JavaScript popolare e ampiamente utilizzato che consente agli sviluppatori di creare applicazioni web interattive e user-friendly in modo rapido ed efficiente. Vue consente agli sviluppatori di creare componenti grafici personalizzati, che si aggiornano automaticamente quando i dati del componente cambiano. Questi componenti possono essere composti insieme per costruire interfacce utente complesse e possono essere facilmente riutilizzati. Oltre al livello di visualizzazione, Vue offre anche una serie di funzionalità che aiutano gli sviluppatori a costruire applicazioni in modo più efficiente. Tra queste, un sistema di data binding semplice e potente, un sistema di dependency injection per la gestione dello stato dell'applicazione e un router per la creazione di SPA (Single Page Application).

[Vite](#): è un development server leggero e veloce per la creazione di applicazioni web moderne. Vite sfrutta il native ES module system per la gestione delle dipendenze; ciò consente di eseguire un rapido ricaricamento dell'applicativo in caso di modifiche al codice e di raggruppare automaticamente le dipendenze su richiesta, senza la necessità di una fase di compilazione separata. Vite supporta anche una serie di funzionalità avanzate, come la suddivisione automatica del codice e il tree-shaking.

[ml-matrix](#): è una libreria TypeScript usata per la manipolazione algebrica di matrici, creata e mantenuta da [Zakodium](#); è un progetto creato da Ecole Polytechnique Fédérale de Lausanne (EPFL) e Université de Lausanne (UNIL) per la creazione di software atto alla manipolazione di dati scientifici, come spettri di massa, dati metabolomici e analisi statistiche. Nello sviluppo del mio progetto ho usato la libreria ml-matrix per



modellare e gestire il concetto di matrici algebriche all'interno del programma.

Quasar: è un framework per la costruzione front-end di applicazioni che utilizzano tecnologie web. È costruito sulla base del framework Vue e fornisce un insieme di strumenti e librerie per agevolare lo sviluppo front-end di alcuni componenti comuni come bottoni, tabelle o layout delle pagine.

TypeScript: è un linguaggio di programmazione open source sviluppato e mantenuto da Microsoft. Si basa su JavaScript e ne estende la sintassi aggiungendo tipi statici opzionali. In JavaScript, le tipologie di dati, non sono definite in modo esplicito e statico quindi, il tipo di una variabile può cambiare durante l'esecuzione del codice. TypeScript introduce i tipi statici, che consentono di specificare in modo esplicito il tipo di una variabile quando viene dichiarata. Ciò aiuta a prevenire errori a runtime comuni, come l'accesso a proprietà di un oggetto che non esiste o che non possiede quella proprietà. TypeScript può essere utilizzato in qualsiasi progetto JavaScript, indipendentemente dalla dimensione o dalla complessità. Per utilizzare TypeScript, è necessario installare il compilatore e configurare il progetto per utilizzare il linguaggio; una volta effettuata questa configurazione è possibile iniziare a scrivere codice TypeScript, che verrà compilato in codice JavaScript per l'esecuzione.

Git: è un software che permette di tenere traccia delle modifiche apportate ai file di un progetto agevolando lo sviluppo e la collaborazione. Il processo di sviluppo di un software utilizzando il programma di gestione delle versioni Git, si può riassumere nel seguente modo: Ogni volta che si desidera fare una modifica ai file di un progetto, si apportano le modifiche e si crea un "commit".

Un commit è una "istantanea" dello stato attuale della repository (cartella contenente tutti i file del programma), esso consiste nel riferimento ai file modificati e in una descrizione delle modifiche apportate. Dopo aver creato un commit con Git, è possibile caricarlo in una repository remota utilizzando i servizi messi a disposizione da alcune piattaforme di sviluppo software come [GitHub](#) o [GitLab](#). Una repository remota è una versione del progetto ospitata su un server remoto, che può essere utilizzata come punto centrale per la collaborazione e la distribuzione del codice. Una volta caricato il commit sulla repository remota, sarà possibile accedere alle modifiche e utilizzare svariati strumenti utili per gestire il codice e la collaborazione, come la gestione delle issue, la creazione di tag e il supporto per la gestione dei branch.

Per maggiori informazioni sul funzionamento della tecnologia fare riferimento al sito ufficiale <https://git-scm.com/>.



# SVILUPPO DEL PROGETTO

Durante la prima fase del tirocinio, mi sono concentrato sullo studio del lavoro di ricerca del professor Frosini, mediante l'articolo [2]. Ciò ha comportato lo studio dei concetti matematici delineati nell'articolo e l'incontro con il professore per eventuali chiarimenti e delucidazioni. Dopo aver acquisito una solida comprensione di questi concetti, ho iniziato la fase di analisi, che prevedeva la progettazione di un software con finalità didattiche. L'obiettivo del software è quello di aiutare gli studenti a comprendere gli strumenti matematici descritti dall'articolo.

Strumenti grazie ai quali dopo un'opportuna selezione di permutanti e funzioni simmetriche è possibile ottenere degli operatori equivarianti non espansivi, ovvero funzioni che non aumentano le distanze tra i dati e sono invarianti sotto l'azione di un gruppo.

Assieme al professore abbiamo deciso di creare un software che permettesse agli studenti di selezionare un gruppo con i relativi permutanti, una funzione in ingresso (un'immagine) e un polinomio simmetrico. Il software elabora questi dati e genera un operatore non espansivo equivariante di gruppo (GENEO). Il risultato dell'elaborazione viene poi visualizzato come immagine, per consentire allo studente, di vedere gli effetti dell'elaborazione del segnale dato in ingresso.

Abbiamo deciso inoltre di visualizzare il dato in ingresso e i risultati dell'elaborazione in una geometria toroidale bidimensionale. Lavorare su un toro, ci ha permesso di avere un dominio compatto, semplificando di conseguenza, il trattamento dei dati.

Per soddisfare le specifiche stabilite dal professore, il software doveva essere leggero e multiplatforma per essere eseguito su una vasta gamma di dispositivi eterogenei, come quelli posseduti dagli studenti, ai quali era



indirizzato l'applicativo. Inoltre il software doveva essere di facile utilizzo e ricco di testi informativi ed esempi.

Sulla base di quest'analisi, ho scelto di creare un'applicazione ibrida utilizzando il framework Tauri. Ho costruito il backend in Rust e ho usato Vue per il front-end. Tauri mi ha permesso di sfruttare la flessibilità e le prestazioni della tecnologia webView gestendo in automatico attraverso i livelli TAO e WRY la compatibilità con i dispositivi. Vue mi ha agevolato nella gestione delle interazioni tra i dati e i componenti grafici, i quali sono stati costruiti usando un altro framework chiamato Quasar, che fornisce componenti grafici standard, modificabili con un certo grado di libertà, molto veloci da integrare e semplici da usare.

Durante il processo di sviluppo è nata la necessità di creare algoritmi in grado di manipolare le matrici per eseguire le trasformazioni richieste sull'immagine in ingresso, come rotazioni, simmetrie e traslazioni. Per questo ho utilizzato la libreria ml-matrix, scritta in TypeScript, per facilitare la modellazione e la trasformazione delle matrici nei miei algoritmi.

Queste trasformazioni sono state applicate su una geometria toroidale bidimensionale, l'uso di un toro bidimensionale per rappresentare un'immagine viene usata per evitare problemi di discontinuità e di bordo. Per mostrare il risultato dell'operatore di gruppo equivariante non espansivo sotto forma di immagine, abbiamo deciso di utilizzare una trasformazione lineare. La trasformazione lineare è stata utilizzata per portare i valori dei canali RGB della matrice risultato, nel range [0,255] mediante la seguente formula:

$$(\text{valore corrente} - \text{valore minimo}) \cdot \frac{255}{(\text{massimo valore} - \text{minimo valore})}$$

È importante notare che il massimo e il minimo valore della matrice, utilizzati nella formula sopra descritta, sono stati presi indiscriminatamente



dal canale dell'immagine, ovvero il massimo e il minimo possono appartenere a canali diversi dell'immagine. Ciò consente di applicare la stessa normalizzazione a ciascun canale, evitando così possibili effetti di disturbo nell'immagine risultante dalla normalizzazione. È importante sottolineare che la normalizzazione non tiene in considerazione il canale alfa, che è stato inserito nella logica del programma per possibili sviluppi futuri. Ciò significa che il programma è potenzialmente in grado di gestire anche immagini con gradi di trasparenza. Attualmente questa funzionalità è ignorata, impostando su tutti gli elementi del canale alfa il valore 255 (trasparenza nulla), al fine di consentire la visualizzazione degli effetti dell'operatore sui canali RGB.

Infine, l'utilizzo della normalizzazione consente di mostrare risultati che altrimenti apparirebbero sempre con un'immagine tutta bianca o tutta nera. Tuttavia, è importante notare che a causa della normalizzazione, non è possibile gestire immagini in cui tutti i canali RGB hanno lo stesso valore, ovvero immagini uniformi come quelle nella scala di grigi da bianco a nero e i colori intermedi.

Un' ulteriore sfida significativa nello sviluppo dell'applicativo è stata la necessità di trasformare il polinomio simmetrico in ingresso, fornito come funzione simmetrica del tipo :

$$\tilde{S}(a_1, \dots, a_n) = \sum_{k_1=0}^{m_1} \dots \sum_{k_n=0}^{m_n} c_{k_1} \dots c_{k_n} \prod_{i=1}^n \sigma_i^{k_i}(a_1, \dots, a_n)$$

in un formato che potesse essere elaborato dal programma.

Ho ottenuto questo risultato applicando i passaggi seguenti:

- Per prima cosa, ho utilizzato delle espressioni regolari sulla stringa in ingresso per semplificarla e rimuovere eventuali informazioni non



necessarie per l'elaborazione. In questo modo, ho potuto preparare la stringa per la successiva elaborazione;

- In seguito, per poter elaborare il risultato, ho scritto un algoritmo che è in grado di scomporre il polinomio passato come ingresso nelle sue parti costituenti, in modo da poter calcolare il contributo di ciascuna di esse al risultato finale. Ad esempio, se il polinomio in ingresso è di questo tipo:

$$\sigma_2(a_1, a_2, a_3, a_4) = a_1 a_2 + a_1 a_3 + a_1 a_4 + a_2 a_3 + a_3 a_4$$

allora l'algoritmo, accettando in ingresso l'insieme delle variabili  $(a_1, a_2, a_3, a_4)$  restituisce tutte le possibili combinazioni di queste variabili in ordine lessicografico crescente, ovvero:

$$[ (a_1, a_2), (a_1, a_3), (a_1, a_4), (a_2, a_3), (a_2, a_4), (a_3, a_4) ]$$

In questo modo, è possibile calcolare il valore di ciascuna combinazione prima di sommarle tutte. Mediante questo algoritmo e le espressioni regolari, sono stato in grado di decomporre la funzione simmetrica nelle sue parti costitutive e di eseguire i calcoli necessari.



# ALGORITMI IMPLEMENTATI

In questo capitolo tratterò nel dettaglio gli algoritmi implementati e le funzioni che questi svolgono.

## PutImageInTorus :

Questo algoritmo ha come obiettivo quello di inserire un'immagine all'interno di un toro, rappresentato come una matrice bidimensionale di dimensioni specificate. Per fare ciò, l'algoritmo, prende in ingresso l'immagine da inserire (image), la larghezza e l'altezza del toro (widthTorus, heightTorus) e le coordinate in cui inserire l'immagine all'interno del toro (xImgInTorus , yImgInTorus).

```
static putImgInTorus( xImgInTorus: number, yImgInTorus: number,  
    image: Matrix, widthTorus: number, heightTorus: number ): Matrix {
```

la funzione inizializza una nuova matrice chiamata torus con le dimensioni specificate heightTorus e widthTorus, riempiendola di zeri. Successivamente calcola la posizione orizzontale e verticale dell'immagine all'interno del toro.

```
widthTorus *= 4;  
var torus = Matrix.zeros(heightTorus, widthTorus);  
  
// Calculates the new horizontal position based on the width of the torus  
xImgInTorus =  
    xImgInTorus < 0  
        ? widthTorus + (xImgInTorus % widthTorus)  
        : xImgInTorus % widthTorus;  
xImgInTorus *= 4;  
  
// Calculates the new vertical position based on the height of the torus  
yImgInTorus =  
    yImgInTorus < 0  
        ? heightTorus + (yImgInTorus % heightTorus)  
        : yImgInTorus % heightTorus;
```



La variabile `xImgInTorus` rappresenta l'indice dei pixel sull'asse `x`, dove inserire l'immagine all'interno del toroide. Se l'indice è inferiore a 0, viene aggiunta la larghezza del toroide (`widthTorus`) per ottenere un indice positivo. Dopo di che, l'indice viene calcolato modulo la larghezza del toroide. Infine, la variabile viene moltiplicata per 4. Questo avviene in quanto ogni pixel dell'immagine per essere visualizzato ha bisogno di 4 valori, RGBA (rosso, verde, blu, alfa).

La funzione quindi effettua un loop attraverso ogni riga e colonna dell'immagine, e per ogni pixel dell'immagine, imposta il pixel corrispondente nella matrice `torus` alla posizione calcolata. Il loop riempie la matrice `torus` a gruppi di quattro colonne alla volta.

```
for (
    var yTorus = yImgInTorus;
    yTorus - yImgInTorus < image.rows;
    yTorus++
) {
    for (
        var xTorus = xImgInTorus;
        xTorus - xImgInTorus < image.columns;
        xTorus += 4
    ) {
        for (var i = 0; i < 4; i++) {
            torus.set(
                yTorus % heightTorus,
                (xTorus % widthTorus) + i,
                image.get(
                    yTorus - yImgInTorus,
                    xTorus - xImgInTorus + i)
            );
        }
    }
}
```

Questo algoritmo viene utilizzato ogni volta che, all'interno del programma, abbiamo la necessità di effettuare uno spostamento lineare di  $(x, y)$  pixel, di modo che l'immagine si “rifletta” nel caso in cui superi i limiti della visualizzazione. Così facendo, è possibile visualizzare in uno spazio finito

segnali che sono virtualmente senza limiti, poiché l'immagine viene "avvolta" intorno ai bordi della visualizzazione invece di terminare bruscamente.

A515 :

La funzione A515 viene utilizzata per restituire una combinazione di ordine lessicografico desiderato, partendo da un gruppo di elementi.

Ad esempio, se si dispone di un insieme di 4 elementi {1, 2, 3, 4} e si desidera selezionare 2 elementi per volta, ci saranno 6 possibili combinazioni: {1, 2}, {1, 3}, {1, 4}, {2, 3}, {2, 4}, {3, 4}. La funzione prende in input la dimensione dell'insieme  $n$ , il numero di elementi da selezionare per ogni combinazione  $p$  e l'indice di ordine lessicografico  $l$  della combinazione desiderata. Se  $n$  è 4,  $p$  è 2 e  $l$  è 2, la funzione restituirà la combinazione di ordine lessicografico 2, ovvero {1, 3}.

```
A515( $n$ : number,  $p$ : number,  $l$ : number) {
```

( la variabile  $l$  verrà scritta in corsivo per non scambiarsela con il numero 1 )

La funzione inizializza un array  $c$  di  $p$  elementi e tre variabili locali:

- $x$ , che rappresenta il primo elemento della combinazione;
- $r$ , che rappresenta il numero di combinazioni possibili di  $p-1$  elementi scelti da un insieme di  $n-x$  elementi;
- $k$ , che viene utilizzata per tenere traccia della posizione corrente nell'ordinamento lessicografico delle combinazioni;

```
var c = new Array(p);  
var x = 1;  
var r = this.binom(n - x, p - 1);  
var k = r;
```



Quindi, viene eseguito un ciclo *while* per trovare il primo elemento della combinazione desiderata. Ad ogni iterazione, *x* viene incrementato di 1 e viene aggiornato il numero di combinazioni possibili di *p*-1 elementi scelti da un insieme di *n*-*x* elementi. Quando *k* supera *l*, ciò significa che l'elemento corrente *x* è il primo elemento della combinazione desiderata.

```
while (k ≤ l) {  
    x += 1;  
    r = this.binom(n - x, p - 1);  
    k += r;  
}  
k -= r;  
c[0] = x - 1;
```

Successivamente, viene eseguito un ciclo *for* per trovare gli altri elementi della combinazione.

```
for (var i = 2; i < p; i++) {  
    x += 1;  
    r = this.binom(n - x, p - i);  
    k += r;  
    while (k ≤ l) {  
        x += 1;  
        r = this.binom(n - x, p - i);  
        k += r;  
    }  
    k -= r;  
    c[i - 1] = x - 1;  
}  
if (p > 1) {  
    c[p - 1] = x + l - k;  
}  
return c;
```

Ad ogni iterazione del ciclo for, viene eseguito il ciclo while interno per trovare l'elemento corrente della combinazione. Quando il ciclo for viene interrotto, viene verificata l'ultima condizione *if*: se *p* è maggiore di 1, viene inserito il valore di  $x+l-k$  nell'ultima posizione dell'array *c*, poiché questo rappresenta l'ultimo elemento della combinazione desiderata.

Questa funzione viene utilizzata per decomporre la funzione simmetrica data in ingresso nelle sue parti costitutive, in modo da poter eseguire i calcoli più agevolmente, come spiegato nel capitolo precedente.

### geneoIn255:

La funzione `geneoIn255` ha lo scopo di preparare una matrice di numeri in modo da poterla visualizzare sullo schermo di un computer. Per fare questo, la funzione mappa l'intervallo di valori presenti nella matrice originale, che può essere molto ampio, nell'intervallo  $[0, 255]$ , che è l'intervallo di valori che può essere visualizzato sullo schermo.

```
var minmax = this.getMatrixMinMax(matrix);  
var M = minmax.max;  
var m = minmax.min;
```

Vengono calcolati il valore massimo (*M*) e il valore minimo (*m*) presenti nella matrice originale. Infine, per ogni elemento (*i*, *j*) della matrice originale, viene calcolato il nuovo valore come segue:

$$([\text{valore matrice alla posizione } i \text{ e } j] - m) * (255 / (M - m))$$

```
for (var i = 0; i < matrixIn255.rows; i++) {  
    for (var j = 0; j < matrixIn255.columns; j++) {  
        if ((j + 1) % 4 == 0) {  
            // alpha channel 255  
            matrixIn255.set(i, j, 255);  
        }  
        matrixIn255.set(i, j, (matrix.get(i, j) - m) * 255 / (M - m));  
    }  
}
```



La porzione di codice mostrata descrive una serie di loop per applicare la trasformazione lineare descritta in precedenza alla matrice dell'immagine, in modo da portare i valori dei canali RGB nell'intervallo  $[0, 255]$ . In particolare, la routine utilizza due cicli for annidati per iterare attraverso ogni elemento della matrice. All'interno di ogni ciclo, viene controllato se l'indice della colonna corrente  $(j) + 1$  è divisibile per 4, ovvero se siamo su un elemento della quarta colonna. Se questa condizione è vera, allora significa che siamo sull'elemento del canale alfa, e quindi l'elemento viene impostato su 255, questo è l'equivalente di impostare la trasparenza a 0, come descritto in precedenza. Successivamente, l'elemento corrente viene modificato tramite la formula per la trasformazione lineare, utilizzando i valori di  $M$  e  $m$  come il massimo e il minimo valore trovato all'interno dell'immagine, il valore corrente viene sostituito con la sua versione normalizzata.

La formula utilizzata per la trasformazione richiede una differenza tra il valore massimo e il valore minimo per essere valida. Se in un'immagine il valore massimo e minimo sono uguali, abbiamo un'immagine monocromatica, senza differenza di tonalità e non si otterrebbe una normalizzazione utile, quindi il programma lancia un'eccezione, che verrà gestita.

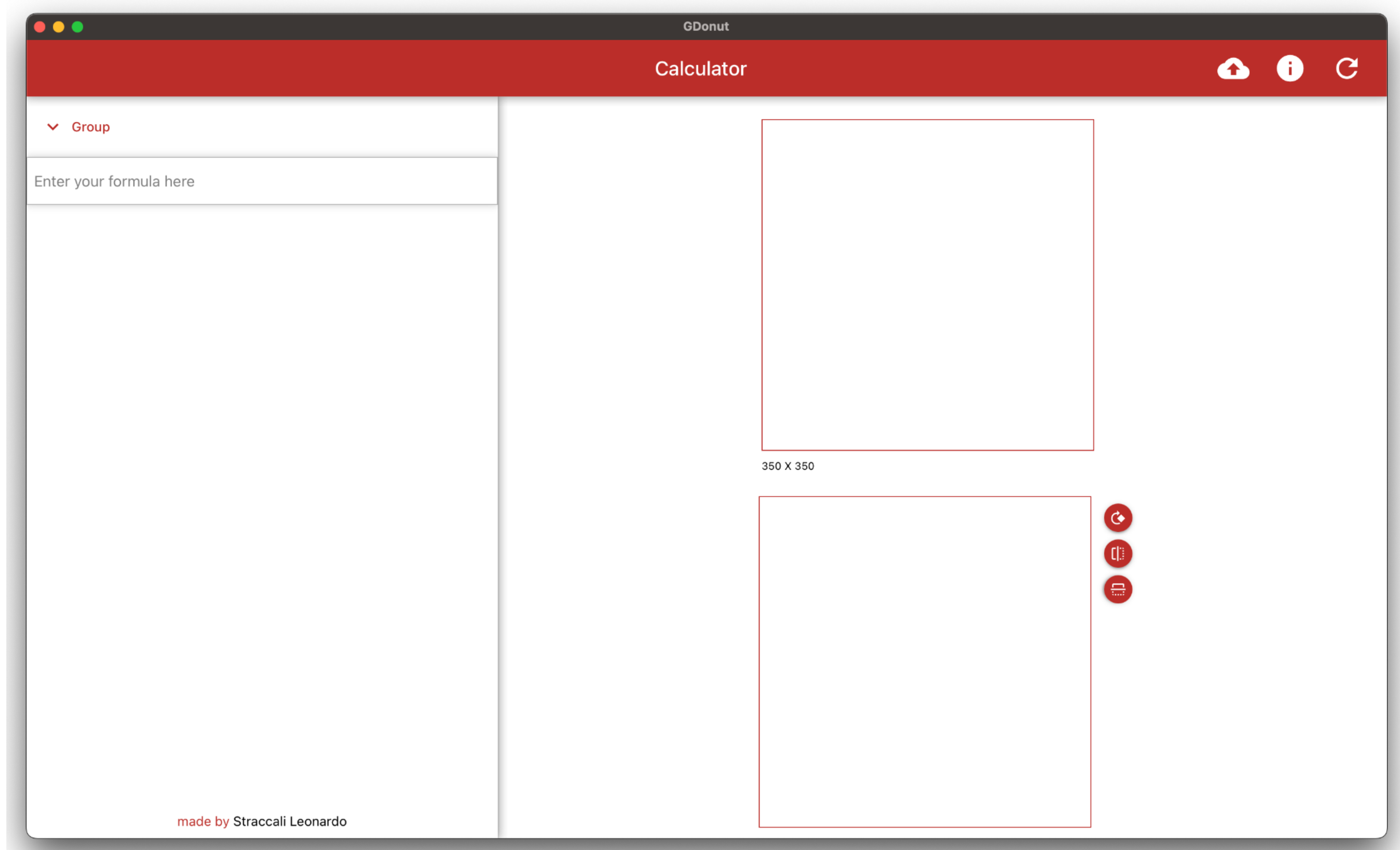
```
if (M == m) {  
    throw new Error("m=M impossible to normalize");  
}
```

Quindi il risultato ottenuto dall'applicativo alla fine dell'elaborazione sarà l'output dell'operatore di gruppo equivariante non espansivo, al quale è stata applicata una trasformazione lineare per portarlo nell'intervallo di valori, utilizzabili per mostrarlo sul display. È importante considerare che il risultato finale è stato sottoposto a questa “normalizzazione”, poiché ciò può influire sull'interpretazione dei dati.

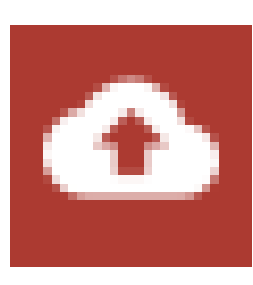




# UTILIZZO DEL PROGRAMMA

Inizialmente, il programma mostra una schermata che contiene una sezione laterale e una centrale, entrambe prive di parametri e immagini.

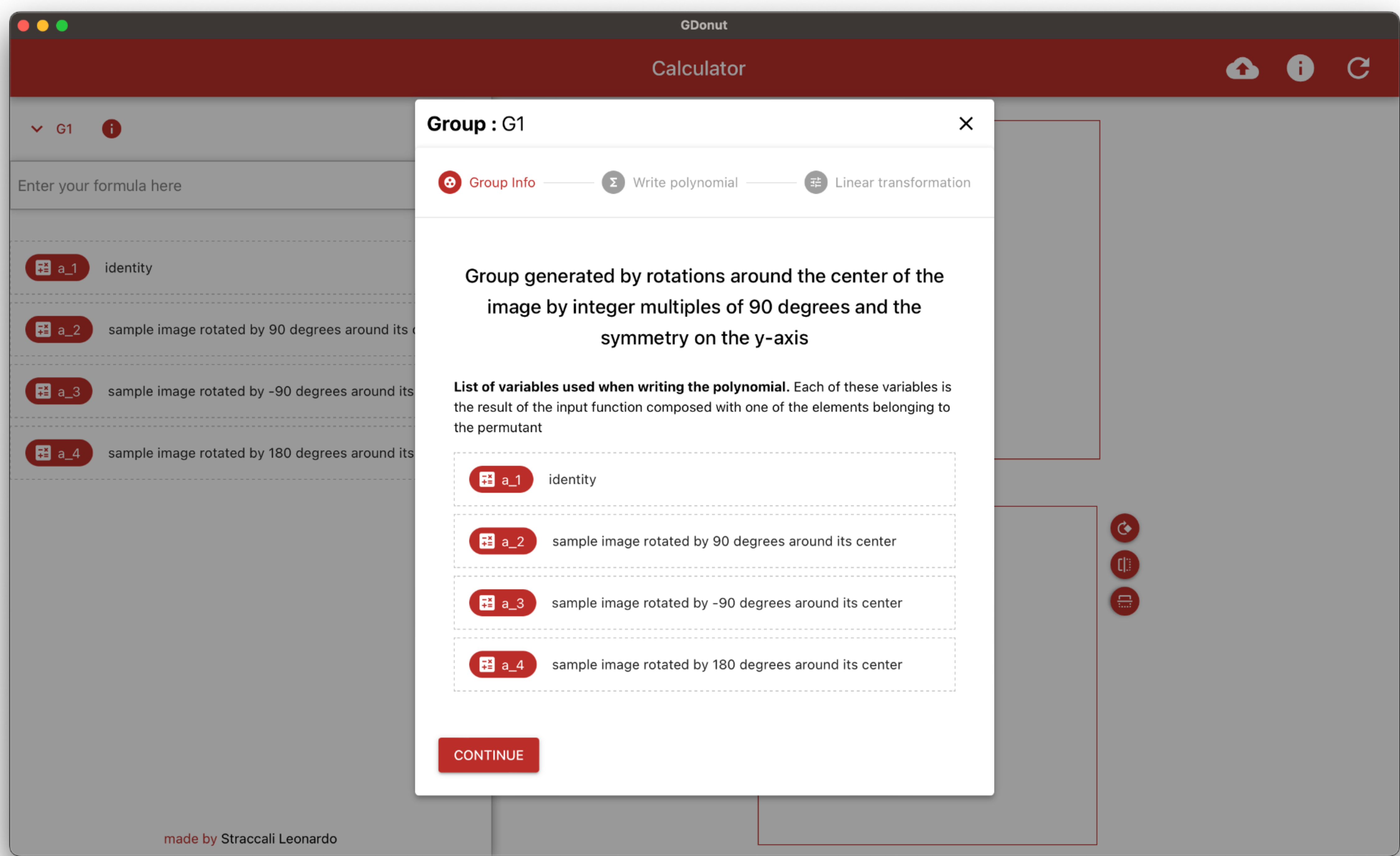


La sezione di sinistra è lo spazio di lavoro dell'utilizzatore del programma: qui è possibile selezionare i gruppi con cui lavorare e immettere il polinomio simmetrico. Nella sezione di destra invece ci sono due riquadri e entrambi sono addetti a mostrare immagini. Il primo in basso con i pulsanti a lato serve per visualizzare l'immagine campione che rappresenterà il segnale di ingresso, mentre il riquadro in alto mostra l'elaborazione dell'operatore di gruppo equivariante non espansivo.

Con il pulsante  in alto a destra, è possibile selezionare l'immagine da caricare dal proprio computer. Mentre i pulsanti accanto sono per ricaricare e pulire lo spazio di lavoro  e per ottenere informazioni sull'utilizzo dell'applicazione .



Quando si sceglie un gruppo, è possibile premere un pulsante informativo accanto al nome del gruppo per ottenere informazioni su di esso. In questa schermata vengono visualizzate informazioni sugli elementi che generano il gruppo e informazioni sul permutante.



Group generated by rotations around the center of the image by integer multiples of 90 degrees and the symmetry on the y-axis

List of variables used when writing the polynomial. Each of these variables is the result of the input function composed with one of the elements belonging to the permutant

a\_1

identity

a\_2

sample image rotated by 90 degrees around its center

Nella seconda sezione della schermata informativa del gruppo, è possibile trovare le istruzioni per scrivere il polinomio simmetrico. Ecco una sintesi di tali istruzioni:

- Usare  $\sigma_n$  come l'n-esimo polinomio simmetrico elementare.
- Usare  $a_n$  come le variabili ottenute dal risultato dell'azione degli elementi dei permutanti sull'immagine in ingresso.
- Nota: non omettere alcun operando, ad esempio il simbolo di moltiplicazione.
- Nota: se si vuole eseguire una divisione, moltiplicare per il reciproco.
- Nota: prestare molta attenzione quando si scrive a pedice degli elementi, perchè potrebbe capitare di continuare involontariamente a scrivere il resto della funzione a pedice, esempio scrivere  $a_{\{1,a_2\}}$  invece di  $a_{1,a_2}$ .

Queste regole spiegano come scrivere il polinomio simmetrico in latex all'interno dell' inputbox che si trova nel pannello a sinistra nella schermata principale.

Write your polynomial in the input box

$$-\sigma_1(a_1, a_2) + 3 \cdot \sigma_2(a_1, a_2)^2 - 6 \cdot \sigma_1(a_1, a_2)$$

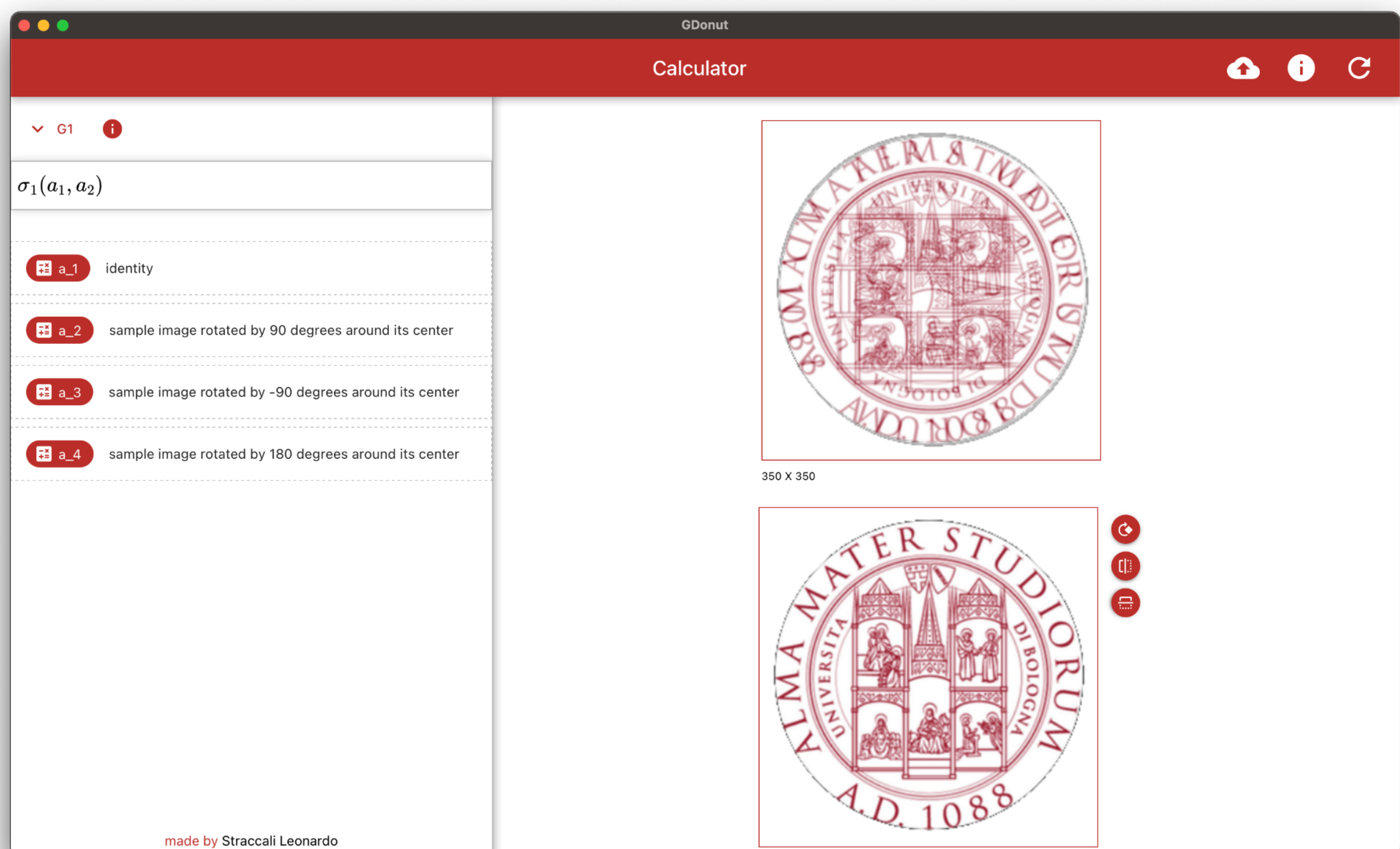
the polynomial must be written in latex

- **Use**  $\sigma_n$  as the n-th elementary symmetric polynomial
- **Use**  $a_n$  as the variables corresponsive to the result obtained from the action of the permutant elements on the input image
- **Note** don't omit any operand such as the \* in multiplication
- **Note** if you want to perform division multiply by the reciprocal
- **Note** be very careful when writing in the subscript of elements because you may unintentionally continue writing the remainder in subscript e.g.  $a_{\{1,a_2\}}$  instead  $a_{1,a_2}$

if the polynomial is formed incorrectly, the geneo canvas will return an error



Dopo aver caricato l'immagine di ingresso e aver scelto il gruppo con cui lavorare, è possibile procedere alla scrittura del polinomio simmetrico nell'inputbox a sinistra. Le modifiche apportate al polinomio verranno immediatamente applicate senza la necessità di premere alcun pulsante e l'elaborazione verrà visualizzata nel riquadro in alto, come mostrato nell'immagine di seguito.



Una volta ottenuta l'elaborazione, è possibile scaricare e salvare l'immagine risultante, facendo clic con il tasto destro del mouse sul riquadro in alto e selezionando l'opzione appropriata; inoltre, è possibile ruotare e riflettere l'immagine di ingresso utilizzando i pulsanti a fianco, nel riquadro in basso, per visualizzare l'effetto sul risultato finale. Queste opzioni possono essere utili per valutare come le modifiche apportate al segnale in ingresso si riflettano sul risultato dell'elaborazione.



## POSSIBILI SVILUPPI FUTURI

L'applicativo è stato sviluppato utilizzando tecnologie innovative e avvalendosi di strumenti supportati da un'ampia comunità di utenti e dotati di una documentazione aggiornata e completa. Il codice sorgente è stato reso disponibile attraverso una repository remota gestita da GitHub, questa soluzione ha permesso di mantenere traccia di tutte le modifiche apportate al progetto, rendendo possibile un eventuale ed ulteriore sviluppo o aggiornamento da parte di altri studenti.

La repository, a cui è possibile accedere al seguente indirizzo (<https://github.com/BabelDev0/GDonut>), è sottoposta alla licenza [MIT](#), che consente di utilizzare, modificare e distribuire il codice sorgente a condizione che venga citato l'autore originale e che non si faccia uso del suo nome per promuovere le modifiche apportate.

Per rendere il codice facilmente comprensibile e agevolare futuri interventi di sviluppo, il progetto è stato completamente commentato e i commit sono stati effettuati seguendo lo standard conventional commit (<https://www.conventionalcommits.org/en/v1.0.0/>). È importante seguire le convenzioni nei commit e nel codice per garantire la coerenza dello stile di codifica all'interno di un progetto. Ciò può contribuire a rendere il codice più facile da comprendere e mantenere nel tempo.

Attualmente, in un'ottica di gestione del progetto, è possibile utilizzare gli strumenti di issue e pull request per segnalare problemi o richiedere nuove funzionalità.



# CONCLUSIONE

Dal punto di vista delle conoscenze apprese, il tirocinio è stato molto valido. Ho avuto l'occasione di studiare e approfondire, in ambito matematico, una ricerca con possibili applicazioni pratiche in campi molto attuali dell'ingegneria informatica, come le reti neurali e l'intelligenza artificiale.

L'obiettivo raggiunto con la fine del tirocinio è stato la creazione di un software per la costruzione di operatori di gruppo equivarianti non espansivi, offrendo uno strumento educativo per gli studenti che desiderano approcciarsi all'argomento.

Grazie a questo tirocinio, ho acquisito nuove competenze tecniche e ho avuto l'opportunità di applicare la mia conoscenza teorica in un contesto pratico. Le nozioni acquisite durante il percorso triennale mi hanno permesso di seguire, con un approccio ingegneristico, il progetto dalle fasi iniziali fino a quelle di sviluppo, creando un applicativo con le caratteristiche di affidabilità, efficienza e manutenibilità.

È possibile scaricare il programma al seguente link:  
<https://babeldev0.github.io/GDonut/>

Ringrazio il mio supervisore per il supporto datomi durante questa esperienza formativa.

# SITOGRAFIA

Nel corso del mio tirocinio, mi sono documentato su diversi siti web per comprendere meglio le tecnologie e gli strumenti utilizzati.

Di seguito riporto l'elenco dei siti web citati nel documento, con i rispettivi link:

- Conventional commit: <https://www.conventionalcommits.org/en/v1.0.0/>
- GDonut: <https://babeldev0.github.io/GDonut/>
- GitHub: <https://github.com/>
- GitLab: <https://about.gitlab.com/>
- Mit license:  
<https://github.com/BabelDev0/GDonut/blob/main/LICENSE>
- ml-matrix: <https://mljs.github.io/matrix/modules.html>
- Quasar.js: <https://quasar.dev/>
- Tauri: <https://tauri.app/it/>
- TypeScript: <https://www.typescriptlang.org/>
- Vite.js: <https://vitejs.dev/>
- Vue.js: <https://vuejs.org/>
- WebKit - WebView: <https://webkit.org/>
- webview2 - WebView:  
<https://learn.microsoft.com/it-it/microsoft-edge/webview2/>
- Zakodium: <https://www.zakodium.com/zakodium>

Il contenuto di questi siti fa riferimento a dicembre 2022, ovvero il momento in cui mi sono documentato su di essi.



# RIFERIMENTI

- [1] Bergomi M. G., Frosini P., Giorgi D., & Quercioli N. (2019). Towards a topological-geometrical theory of group equivariant non-expansive operators for data analysis and machine learning. *Nature Machine Intelligence*, 1, 423–433.
- [2] Conti, F., Frosini, P., & Quercioli, N. (2022). On the Construction of Group Equivariant Non-Expansive Operators via Permutants and Symmetric Functions. *Frontiers in Artificial Intelligence*, 5.