DESIGN DI UN SISTEMA DI CARATTERIZZAZIONE DEGLI ATTUATORI DI UN DRONE

BERTUZZI FEDERICO
GERSID HYKA
BRUNO ZANIN

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Presentazione del Progetto

Il progetto si sviluppa su un banco prova che permetta di testare il funzionamento e le caratteristiche di attuatori, composti da motore ed elica, che risultano parte fondamentale nella realizzazione dei droni.

Gli obiettivi sono:

- Migliorare le prestazioni dell'intero design in termini di stabilitá;
- Migliorare la compatibilitá nella comunicazione tra parte software e hardware;
- Garantire affidabilitá dei risultati ottenuti.

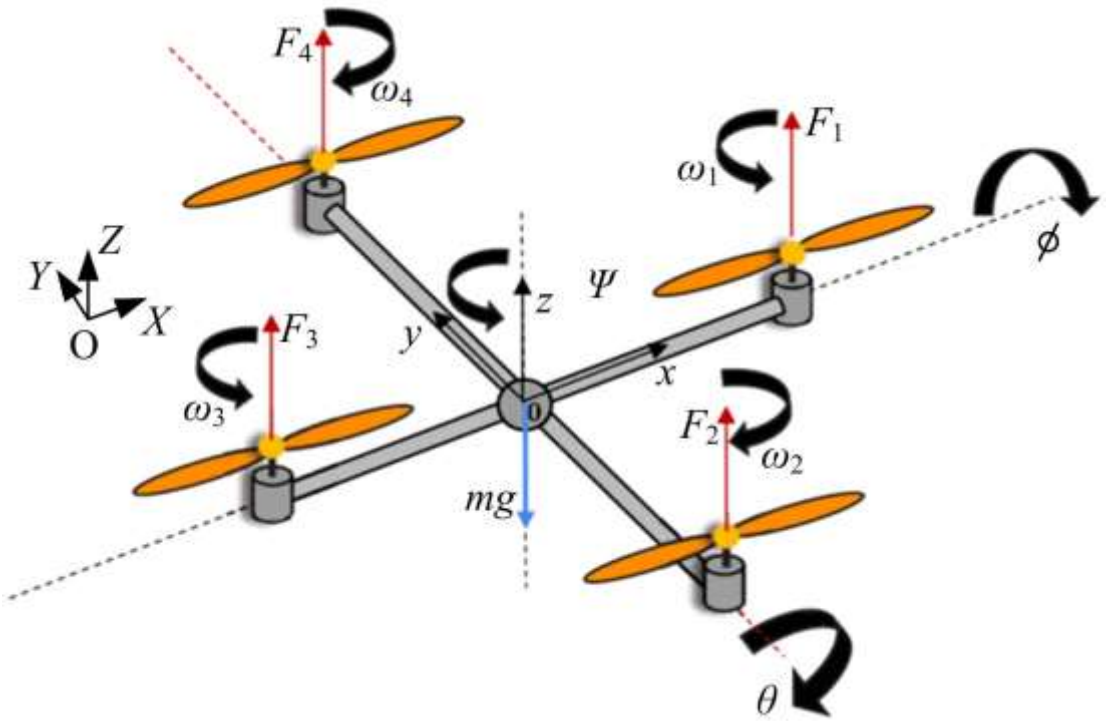
DRONI: AEREI SENZA PILOTA

UNIVERSITÀ DEGLI STUDI DI PADOVA

Un drone è un **veicolo aereo autonomo** senza equipaggio (UAV) **controllato** da un **sistema informatico** a distanza o a bordo.
La definizione fa luce su tre aspetti:

- Dinamica del veicolo: insieme di grandezze (forze e velocitá) che interagiscono tra loro al fine di ottenerne altre (coppie);
- Sistema di controllo: grandezze in output gestite tramite variazioni delle grandezze in input;
- Sistema informatico: sistema di elaborazione dati gestito a lato software.

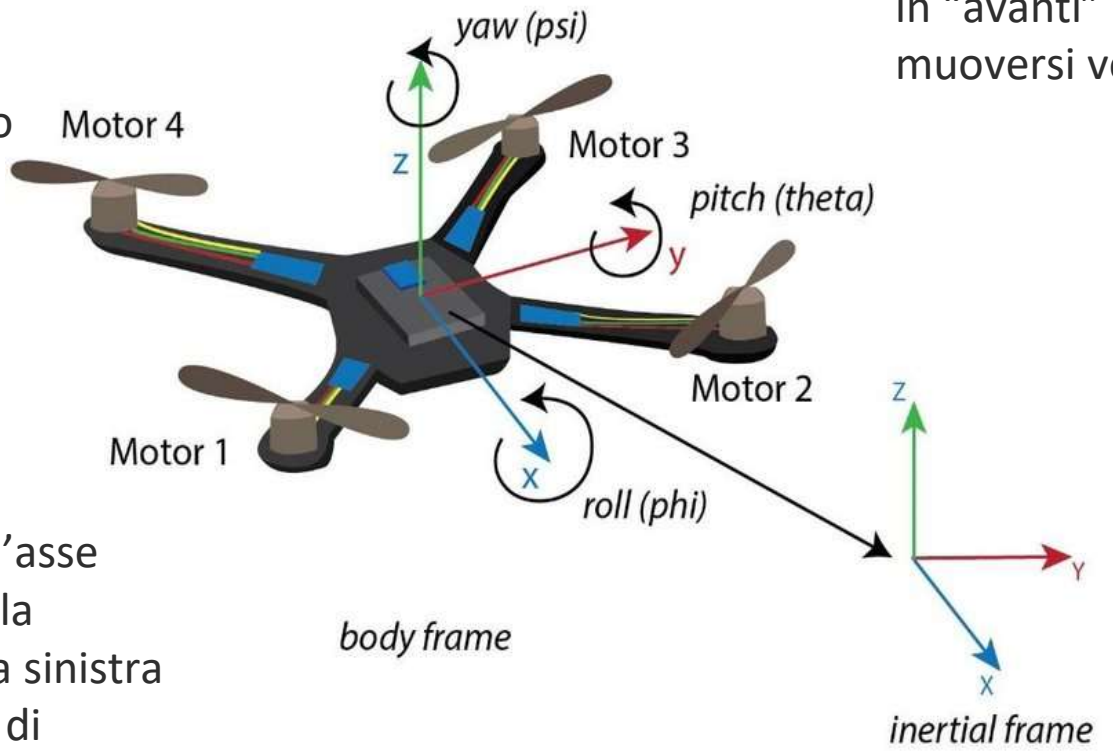
PRINCIPIO DI FUNZIONAMENTO



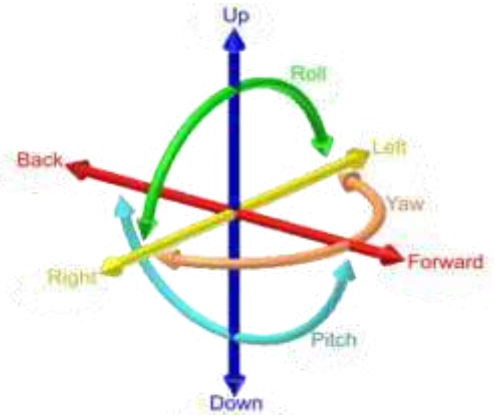
Affinchè il drone voli, è necessario un sistema che sia in grado di generare una forza opposta alla **forza di gravità**, e generare una **spinta** dal basso verso l'alto superiore al peso al decollo del quadricottero.
Una forza costante che nel caso dei quadricotteri è data dalla rotazione di quattro eliche .
Ogni elica è azionata da un motore e possono roteare ad una velocità differente e indipendente dagli altri tre. I tre movimenti principali di un drone sono **yaw**, **pitch** e **roll** possono avvenire solo con un controllo accurato delle velocità dei motori e quindi delle rotazioni delle eliche relative il cui senso di rotazione, orario o antiorario, dipenderà dal tipo di spostamento desiderato.

PRINCIPIO DI FUNZIONAMENTO

Yaw è la rotazione sull'asse verticale, e permette appunto la rotazione a 360° del quadricottero sul proprio asse.



Pitch è la rotazione sull'asse trasversale, ovvero la rotazione in "avanti" ed "indietro" per muoversi verso queste direzioni.



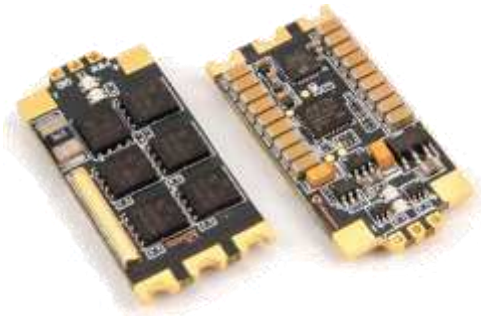
Roll è la rotazione sull'asse longitudinale, ovvero la rotazione verso destra sinistra che permette al quad di muoversi verso queste due direzioni.

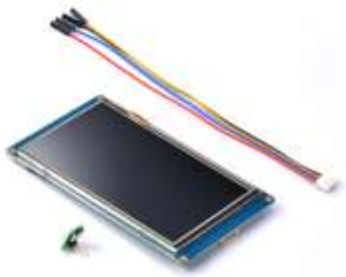
DISPOSITIVI E SOFTWARE UTILIZZATI

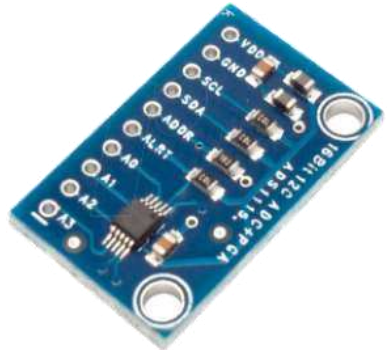
UNIVERSITÀ DEGLI STUDI DI PADOVA

DISPOSITIVI

- Brushless Motor : Holybro 2206-kv2300
- ESC: Tekko F4 35 A
- Celle di Carico: TAL220
- Arduino Mega
- Display Nextion
- 16-bit ADC





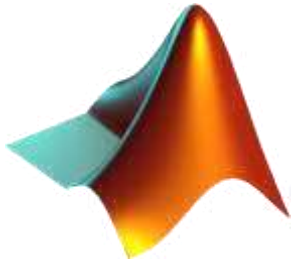


DISPOSITIVI E SOFTWARE UTILIZZATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

SOFTWARE

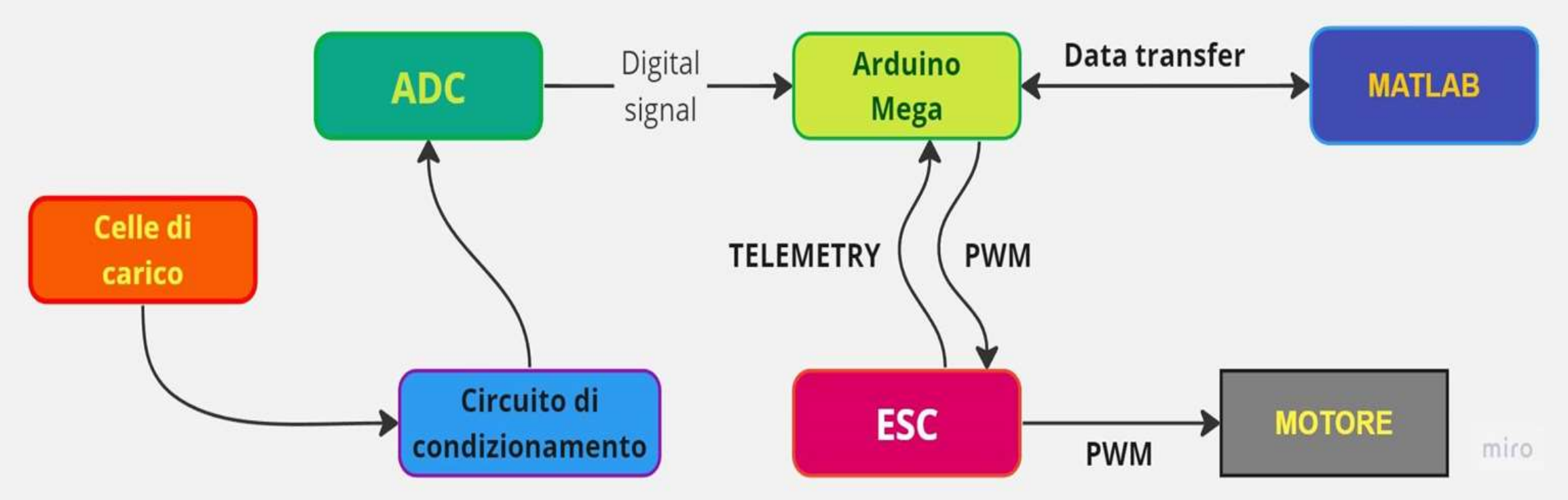
- Matlab
- Arduino Ide
- Kicad
- EasyEDA
- Blhelisuite32
- Autodesk Inventor
- Lubanse
- Falstad



BANCO PROVA: PRESENTAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

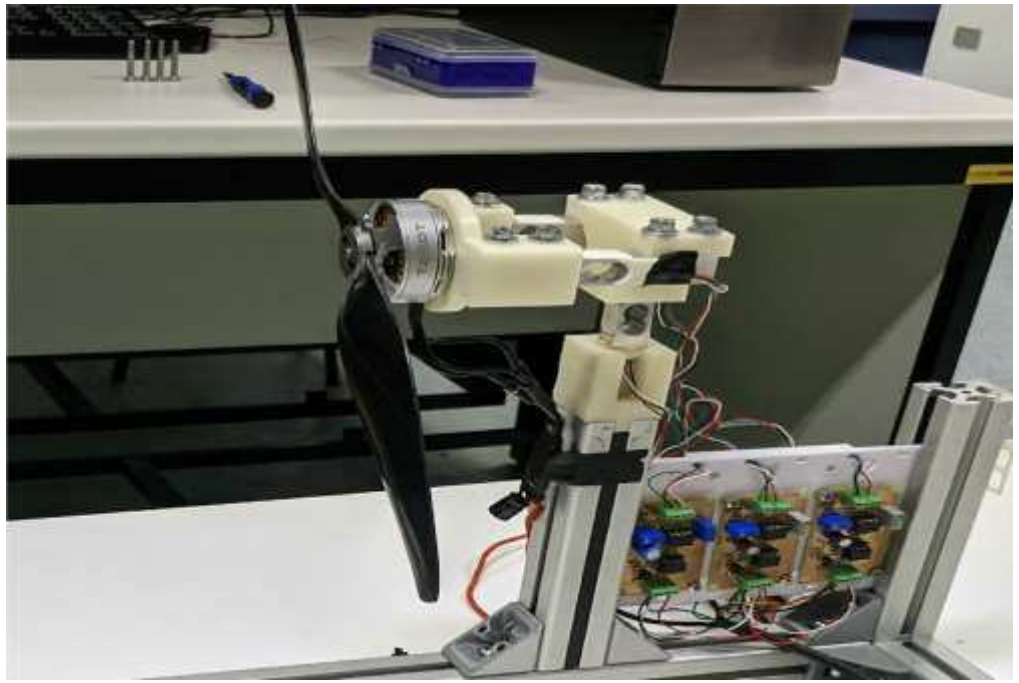
LAYOUT E FUNZIONAMENTO

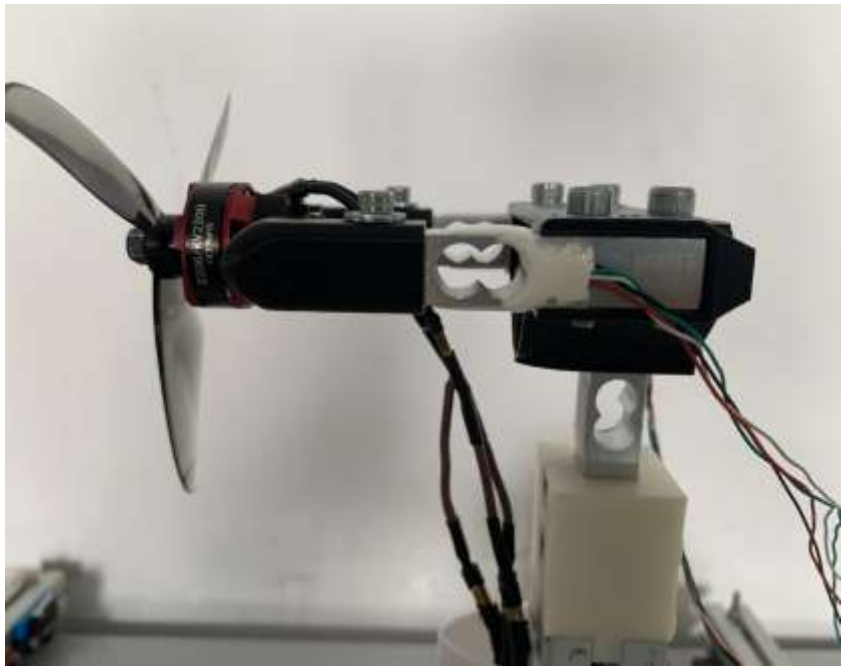


BANCO PROVA: SUPPORTI

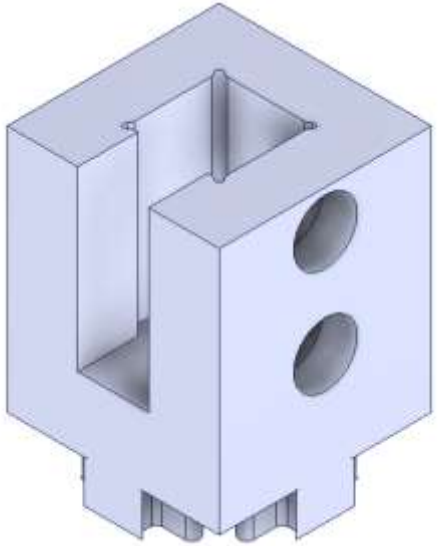
UNIVERSITÀ DEGLI STUDI DI PADOVA

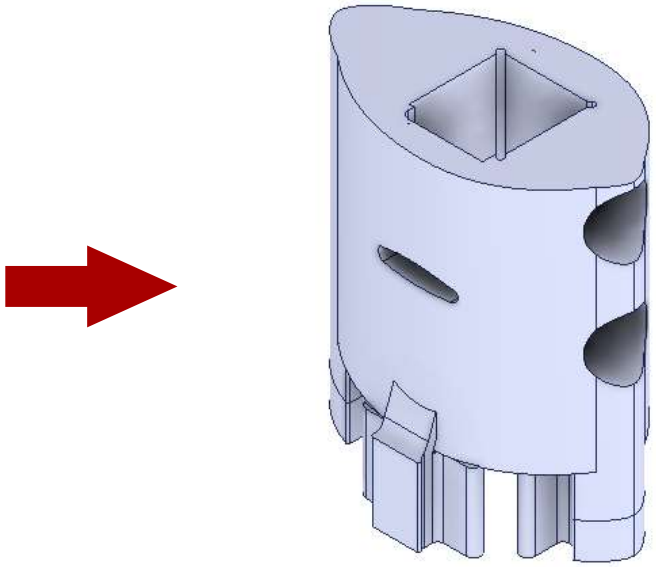
- Supporti progettati con Autodesk Inventor e realizzati attraverso la stampa in 3D con filamento in PLA.
- Utilizzo di geometrie più complesse per ridurre la sezione frontale e favorire il flusso d'aria prodotto dall'elica.
- Realizzazione di diversi prototipi per cercare la soluzione migliore



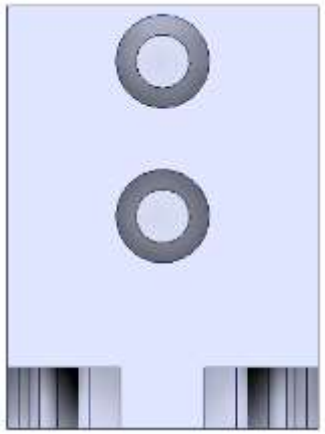


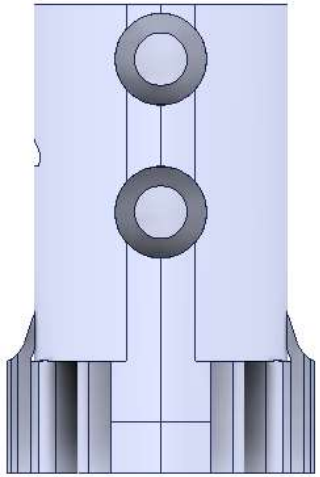
BANCO PROVA: SUPPORTI

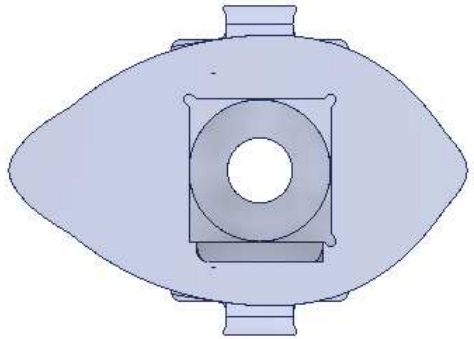




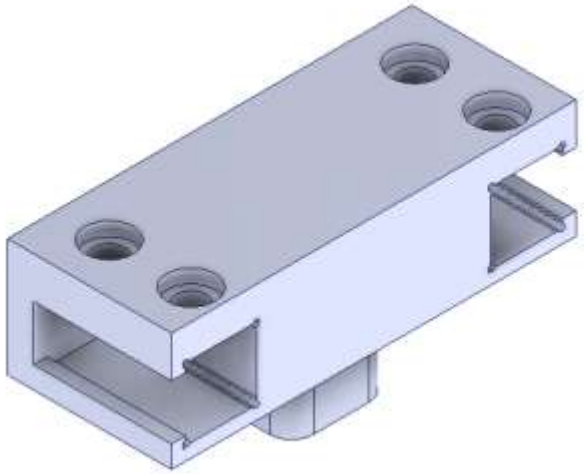
SUPPORTO CELLA VERTICALE

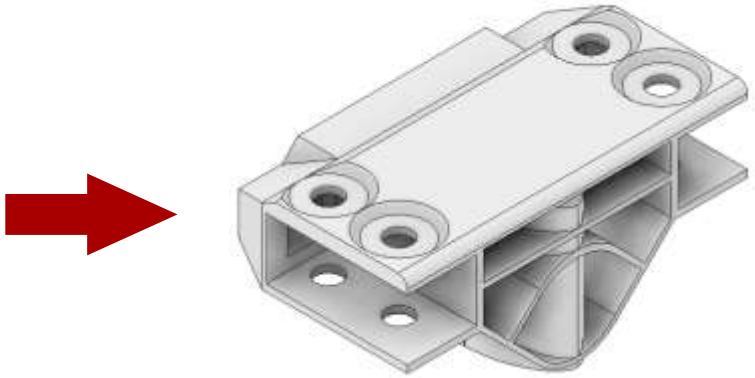




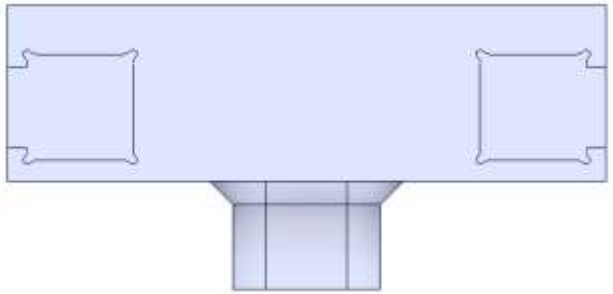


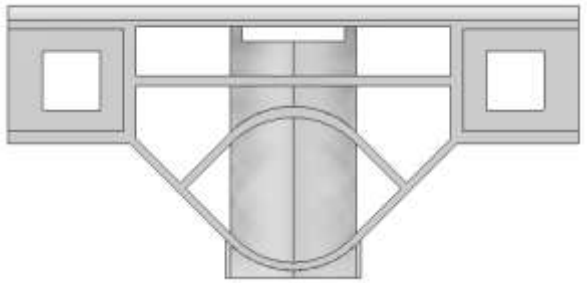
BANCO PROVA: SUPPORTI

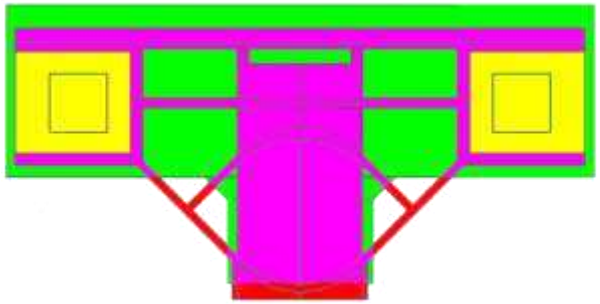




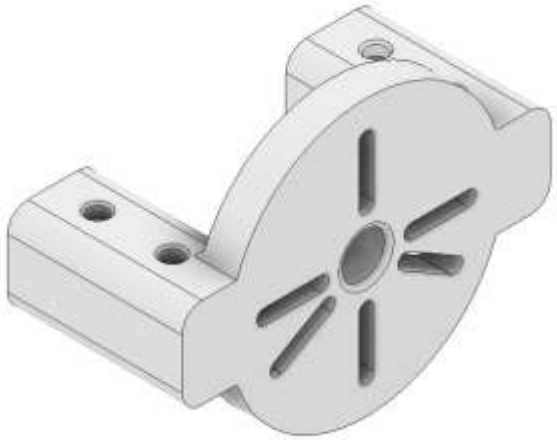
SUPPORTO CELLE ORIZZONTALI

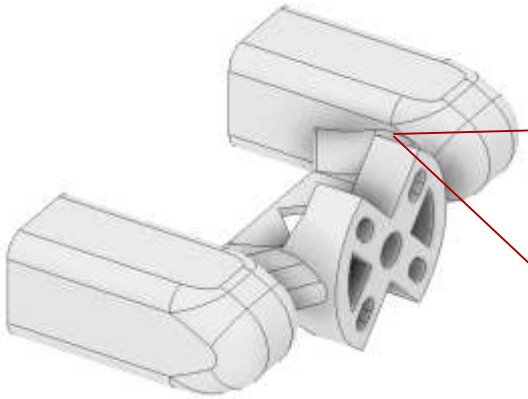




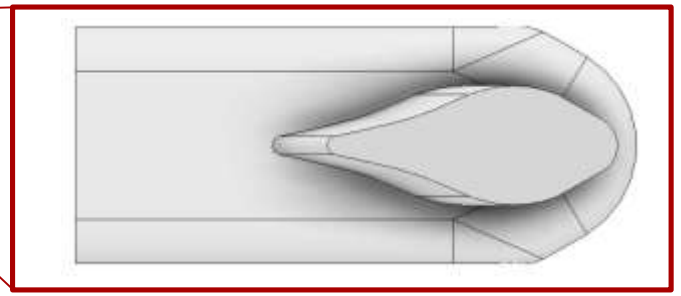


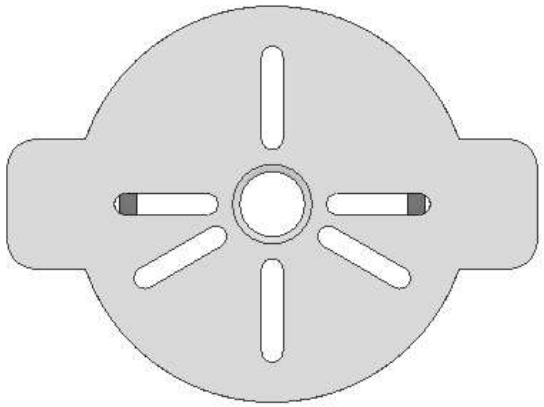
BANCO PROVA: SUPPORTI

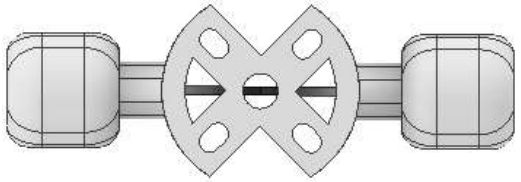


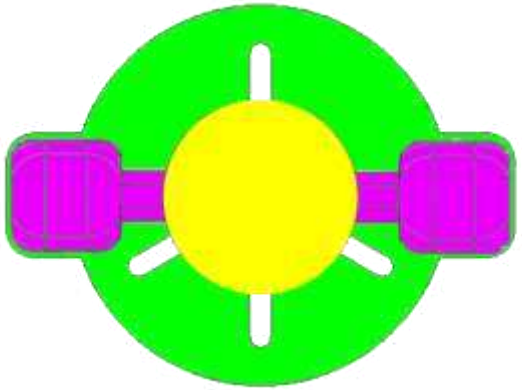


SUPPORTO MOTORE

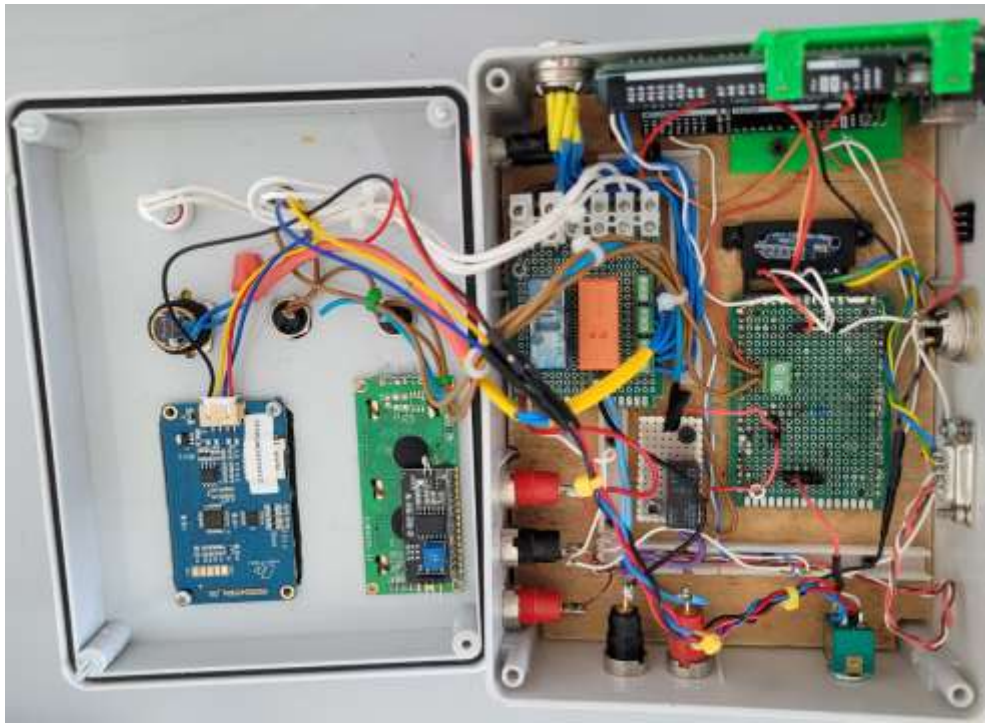




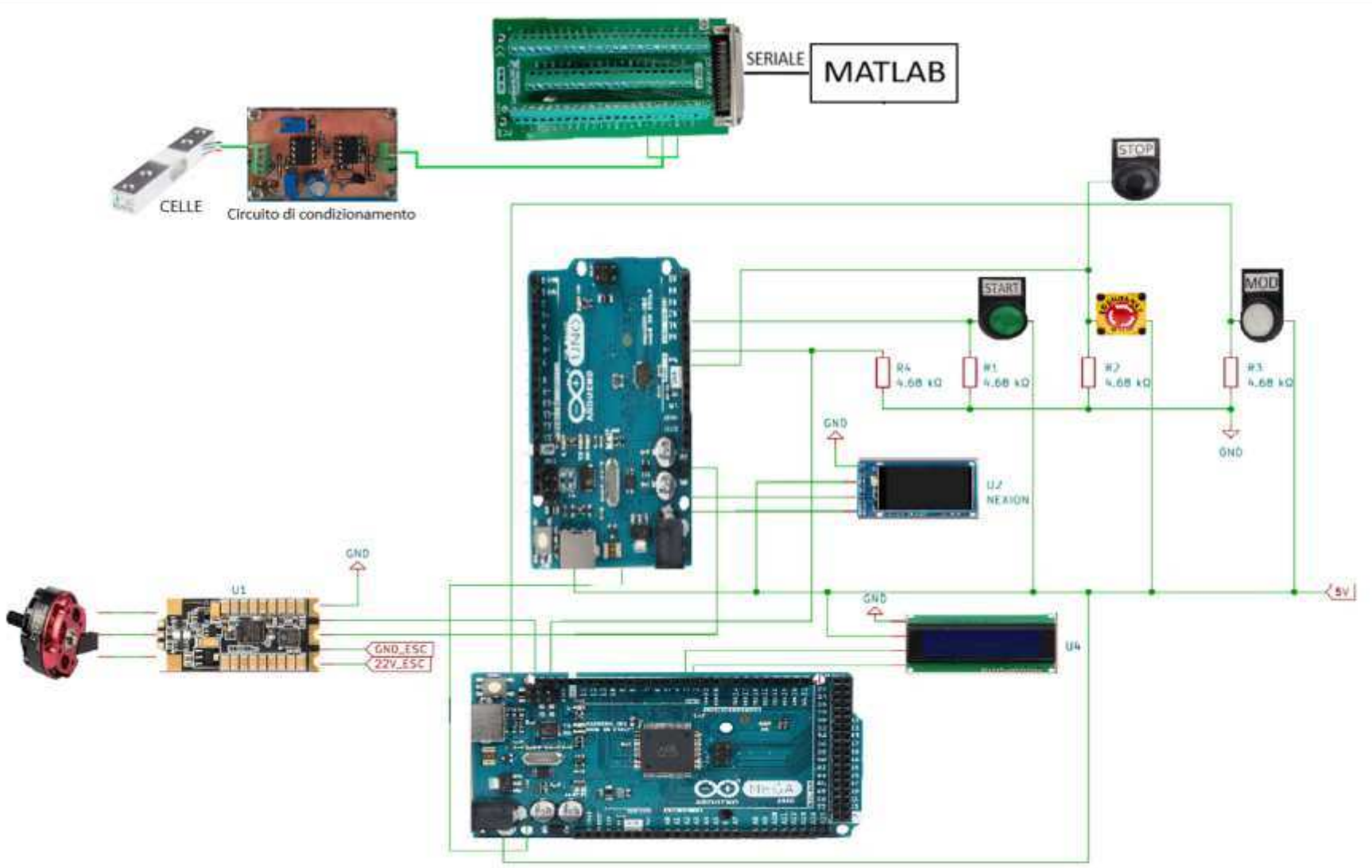




QUADRO ELETTRICO



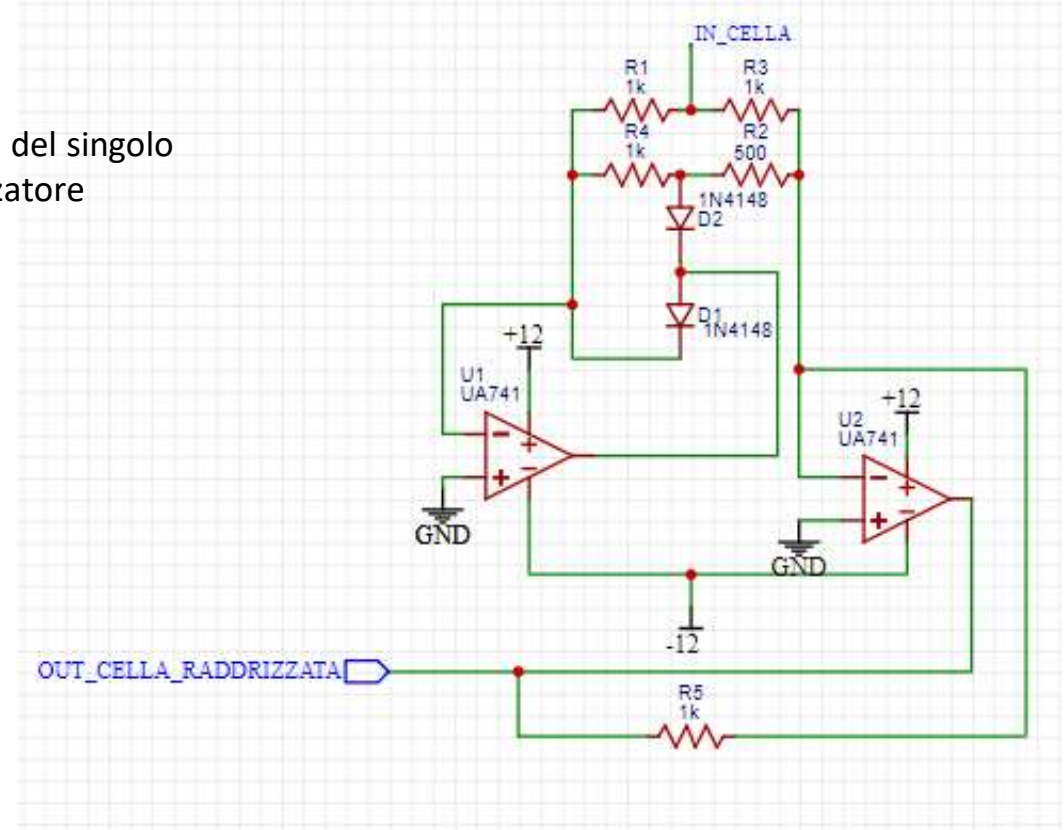
QUADRO ELETTRICO: SCHEMA (PRIMA)

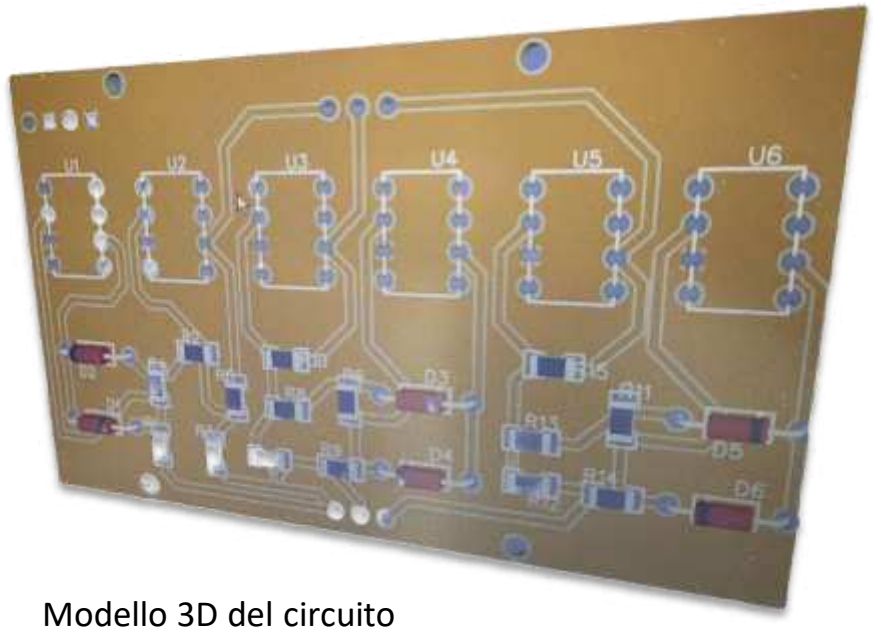


QUADRO ELETTRICO: CIRCUITO RADDRIZZATORE

UNIVERSITÀ DEGLI STUDI DI PADOVA

Schema del singolo raddrizzatore

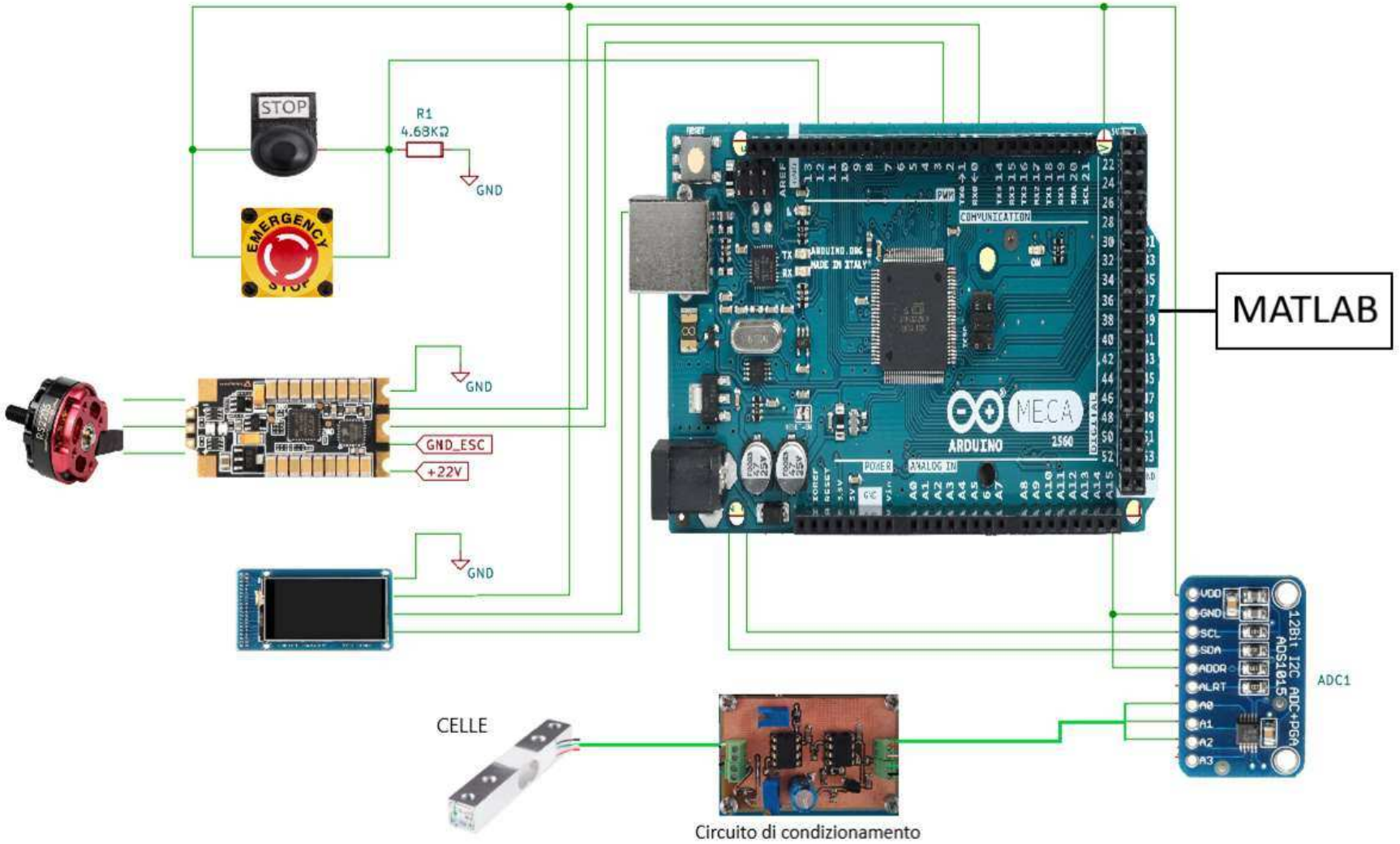




Modello 3D del circuito stampato

Il circuito realizzato è composto da tre raddrizzatori a doppia semionda. In ingresso sono presenti i segnali provenienti dalle celle, mentre in uscita le tensioni raddrizzate andranno all'Arduino.
Soluzione non è stata adoperata in favore di ADS 1115.

QUADRO ELETTRICO: SCHEMA (DOPO)



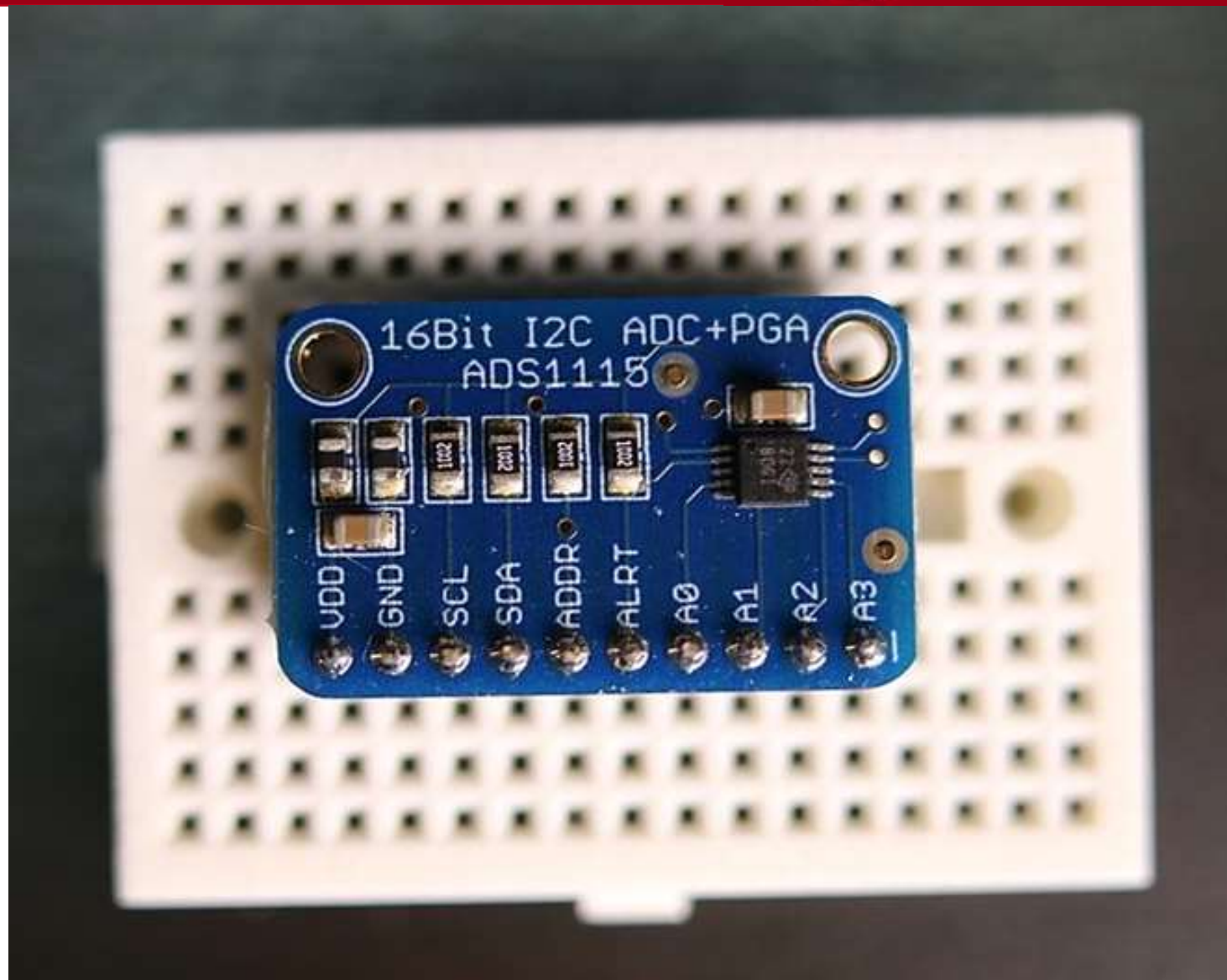
QUADRO ELETTRICO: CARATTERISTICHE ADC

UNIVERSITÀ DEGLI STUDI DI PADOVA

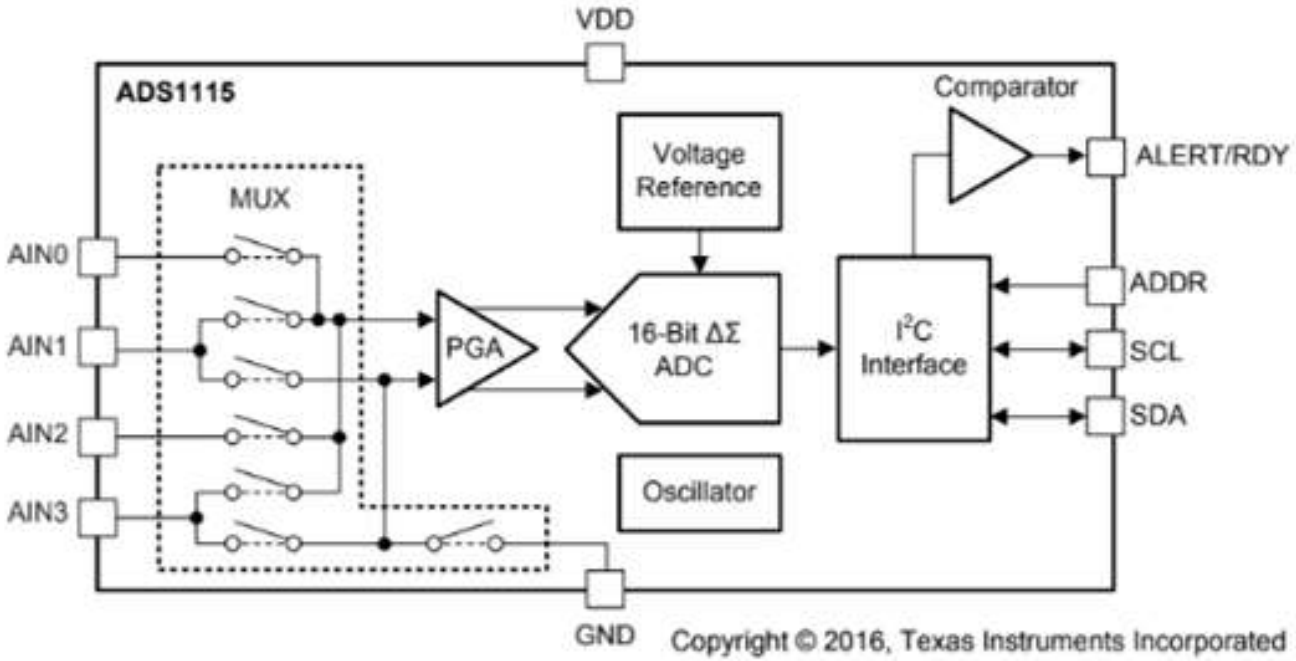
Per la conversione del segnale da analogico si è deciso di utilizzare un convertitore esterno a 16 bit.

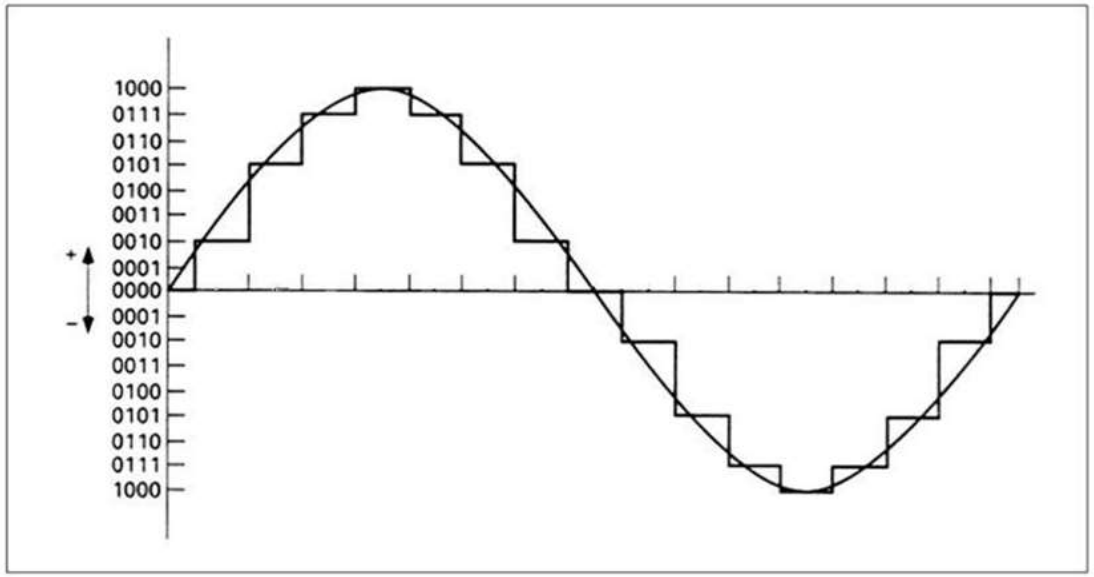
Questa soluzione risulta migliore perché:

- **Meno componenti** rispetto al progetto del raddrizzatore (più affidabilità);
- Risulta **più piccola** (meno ingombro nel quadro elettrico);
- **Risoluzione maggiore** rispetto all'ADC già presente in Arduino Mega (16bit vs 10bit);
- Range di **acquisizione** è **variabile** (rispetto a quello di Arduino che è fisso da 0 a 5V).



QUADRO ELETTRICO: FUNZIONAMENTO ADC

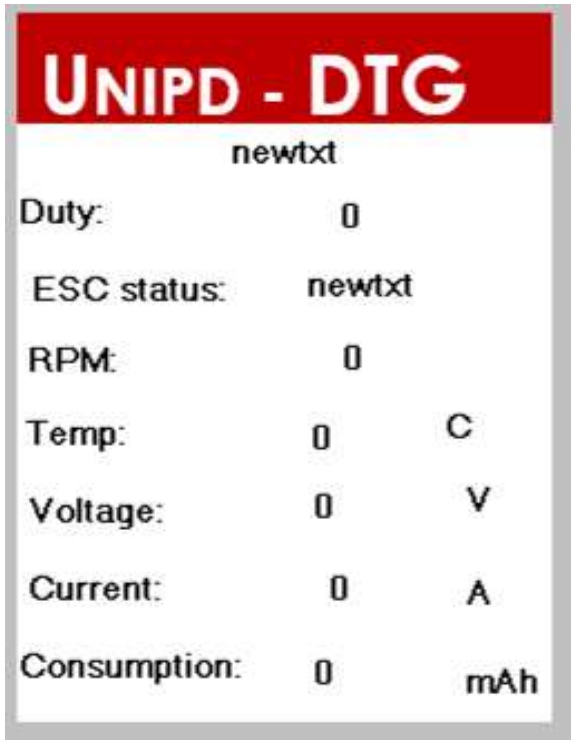




A monte dell'ADC, il **PGA** (*Programmable Gain Amplifier*) consente di regolare il guadagno dell'ampiezza del segnale analogico in modo da sfruttare al meglio la gamma dinamica per la misurazione. (**ads.setGain(GAIN_ONE); //1x gain +/- 4.096V 1 bit= 2mv 0.125mV**)

L'ADC è in grado di comunicare con Arduino tramite librerie preesistenti e **I2C** (Inter-Integrate Circuit).

QUADRO ELETTRICO: SCHERMO NEXTION

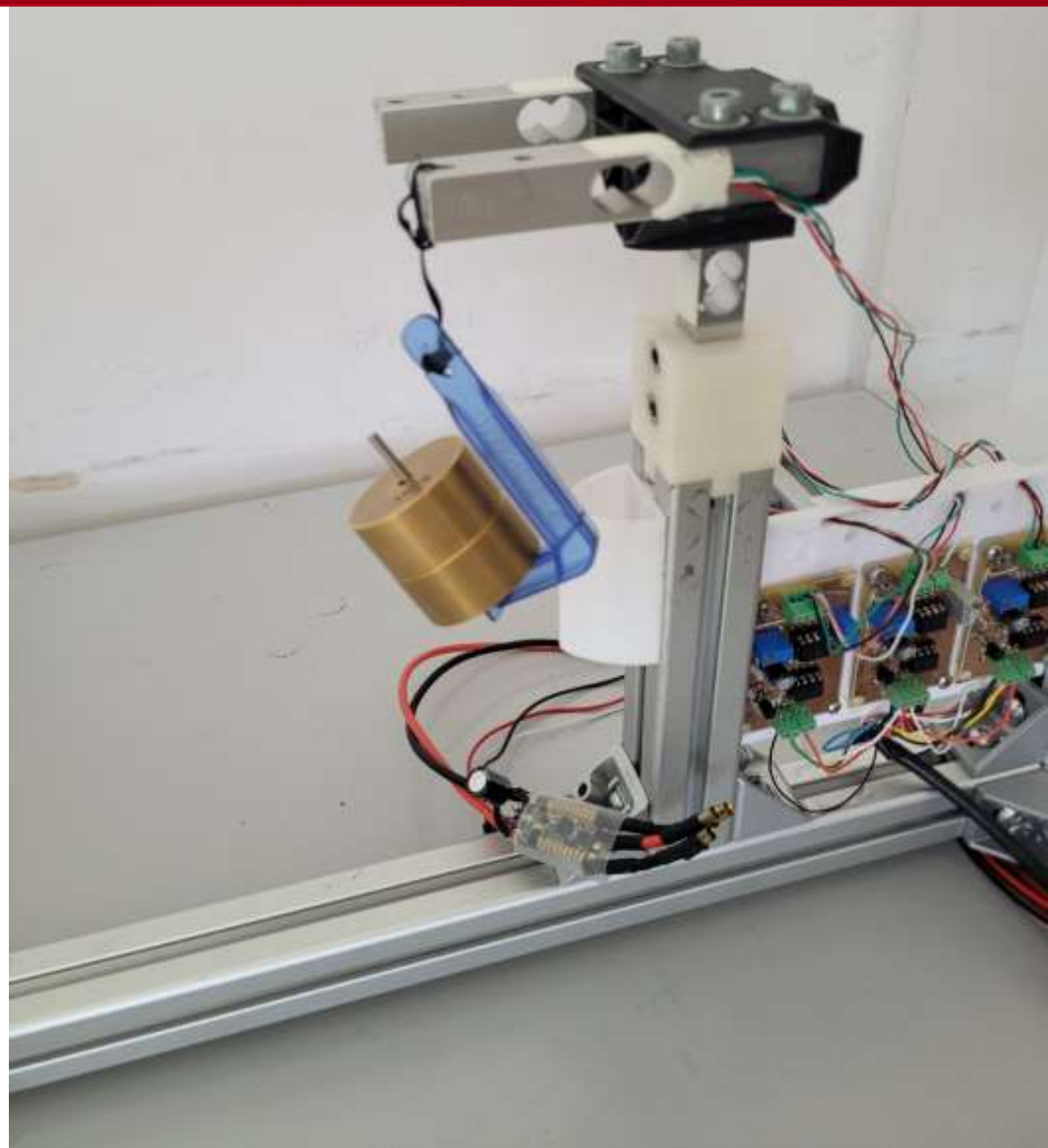


Si è utilizzato un unico display per la visione dei dati come telemetria, consumo, duty cycle, tensione e velocità.

CODICI: TARATURA CELLE DI CARICO

UNIVERSITÀ DEGLI STUDI DI PADOVA

```
%% PARAMETERS

load('K_parameters.mat', 'K');
K0=K(1,:)
K1=K(2,:)
K2=K(3,:)
K_update = zeros();
mass = [0.005 0.105 0.205 0.305 0.205 0.105 0.005];
force = mass*9.81
step_quantity = numel(mass) % # of steps
sample_quantity = 1000;      % # of samples for each step

acquired_values = zeros(step_quantity , sample_quantity);
```



CODICI: TARATURA CELLE DI CARICO

UNIVERSITÀ DEGLI STUDI DI PADOVA

CONNESSIONE AUTOMATICA

```
available_ports = serialportlist("available")
id_word = 'hola';
size = numel(available_ports);
for n=1:size
    s = serialport(sprintf('%s', available_ports(n)), 115200);
    in_word = read(s, 4, 'char');
    if strcmp(id_word,in_word)==0
        write(s,'n','char');
        flag = 0;
    end

    if strcmp(id_word,in_word)==1
        write(s, 'y', 'char');
        port_number=n;
        disp(port_number);
        flag = 1;
        break
    end
end

if flag==1
    disp("Succesfully connected to: ");
    disp(sprintf('%s', available_ports(port_number)));
end

write(s,'c','char')
```

```
void setup() {

  pinMode(PWM_PIN, OUTPUT);
  pinMode(STOP_PIN, INPUT_PULLUP);
  //INITIALIZE SCREEN SERIAL
  screen_serial.begin(115200);

  //INITIALIZE ESC
  ESC_serial.begin(115200);
  ESC_serial.listen();

  setPwmFrequency(PWM_PIN, divisore);

  analogWrite(PWM_PIN,0);

  //AUTOMATIC PORT CONNECTION (WORKING)
  beg:
  Serial.begin(115200);
  while(!Serial){

  }
  Serial.write("hola");
  while(!Serial.available()){
  }
  message = Serial.read();
  if(message=='n'){
    send_info="t16.txt=\"Problem\"";
    writeString(send_info);
    Serial.end();
    goto beg;
  }
 if(message=='y'){
  send_info="t16.txt=\"conn\"";
    writeString(send_info);
 }
  //set program to execute
  while(!Serial.available()){

  }
  message = Serial.read();
  //ADC SETTING AND VERIFY

  adc.setGain(GAIN_ONE);
  delay(1000);
  if (!adc.begin()) {
    send_info="t16.txt=\"ADS_DOWN\
    writeString(send_info);
    while (1);

  }

}
```

CODICI: TARATURA CELLE DI CARICO

UNIVERSITÀ DEGLI STUDI DI PADOVA

ACQUISIZIONE DATI

```
disp("Press when ready.");
pause
cell_number = input("Select the cell (0,1,2):");
write(s,cell_number,'char')
acquired_values = zeros(step_quantity,sample_quantity);
plot_values = zeros(sample_quantity*step_quantity,1);
i=1;
j=1;
while i<=step_quantity
    disp("Desired weight:")
    mass(i)
    disp("Press when ready.")
    pause
    write(s,'q','char');
    while j<=sample_quantity
        j
        bytes = read(s, 2,'uint8');
        acquired_values(i,j) = typecast(uint8(bytes(1:2)), 'int16');
        plot_values(j + (i-1)*sample_quantity , 1)=acquired_values(i,j);

        j=j+1;
    end
    j=1;
    i=i+1;
end

 figure(2);
grid on;
title('values from adc');
plot( plot_values );
```

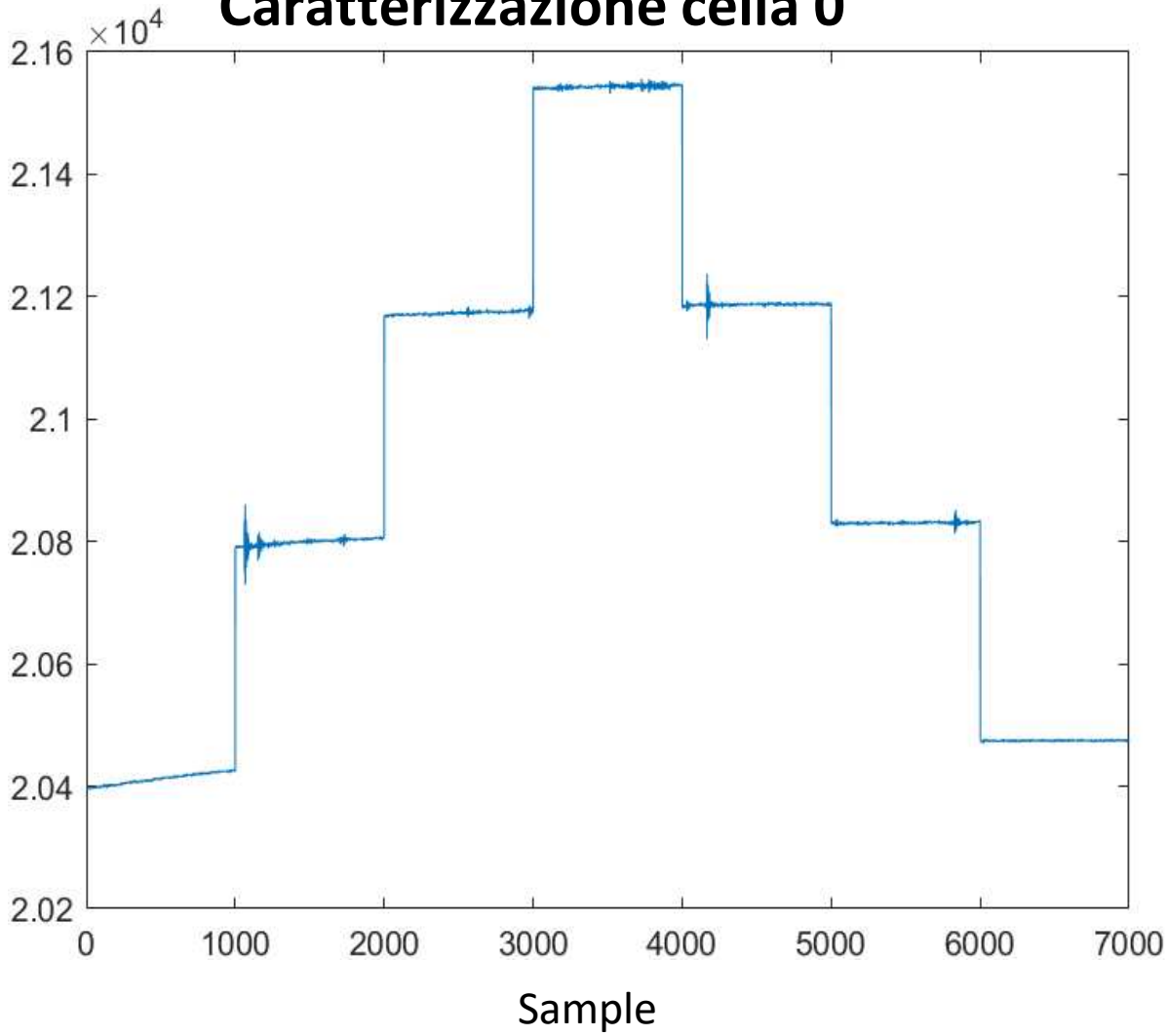
```
if(message=='c'){
  send_info="t16.txt=\"Charact\"";    //writeString send str
  writeString(send_info);
  String serial_input;
  while(!Serial.available()){

  }
  serial_input = Serial.read();
  cell_number = serial_input.toInt();
  i=0;
  j=0;
  while(i<sizeof(duty_start)){
      while(!Serial.available()){}
      Serial.read();
      while(j<sample_quantity_charact){
        cell_value = adc.readADC_SingleEnded(cell_number);
        byte *pointer=(byte *) &cell_value;
        Serial.write(pointer, 2);
        delay(12);
        j++;
        ////////////////////////////////////////////////

      }
      j=0;
      i++;
    }
    i=0;
    delay(10);

}
```

CODICI: TARATURA CELLE DI CARICO

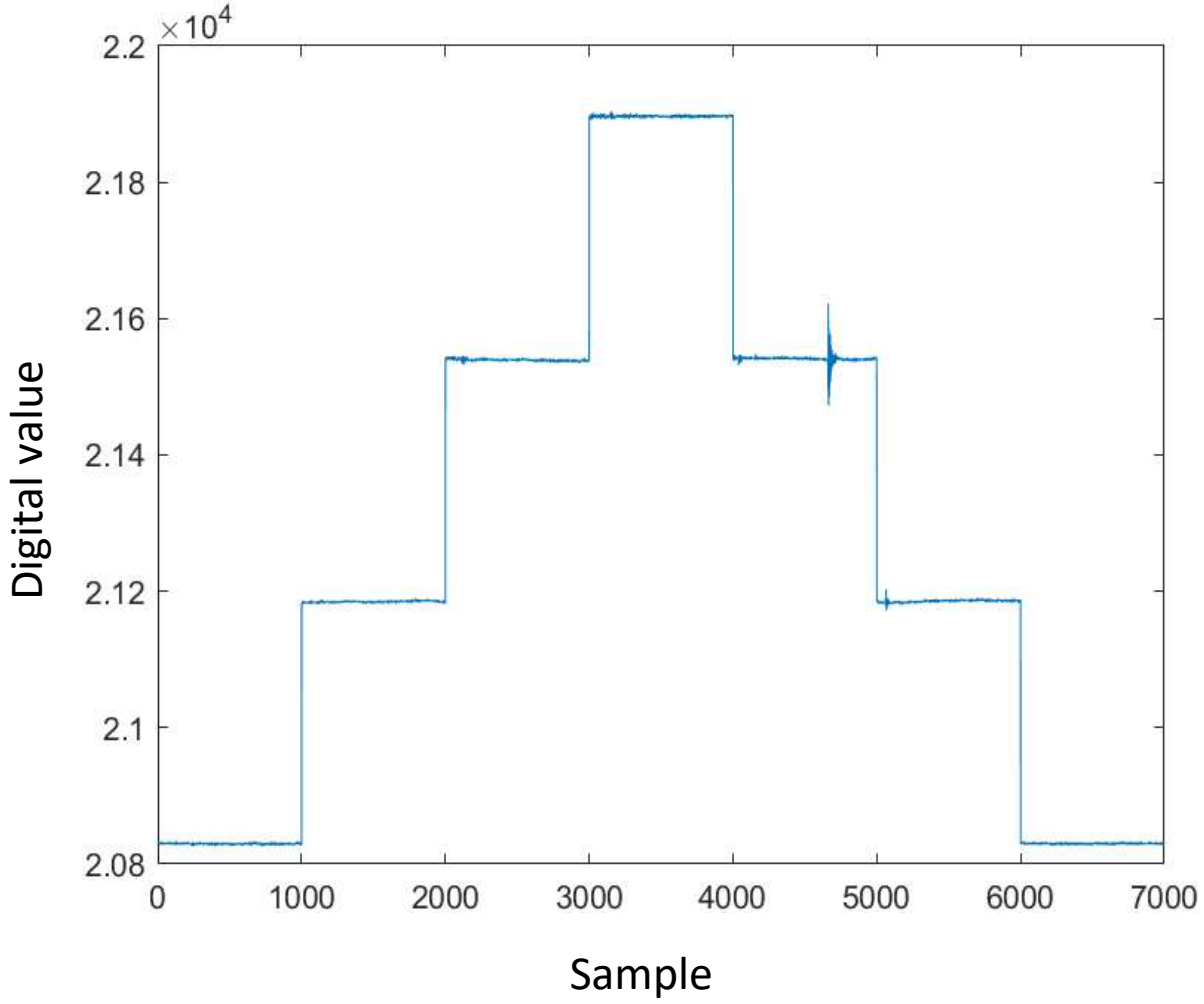
UNIVERSITÀ DEGLI STUDI DI PADOVA

GRAFICI

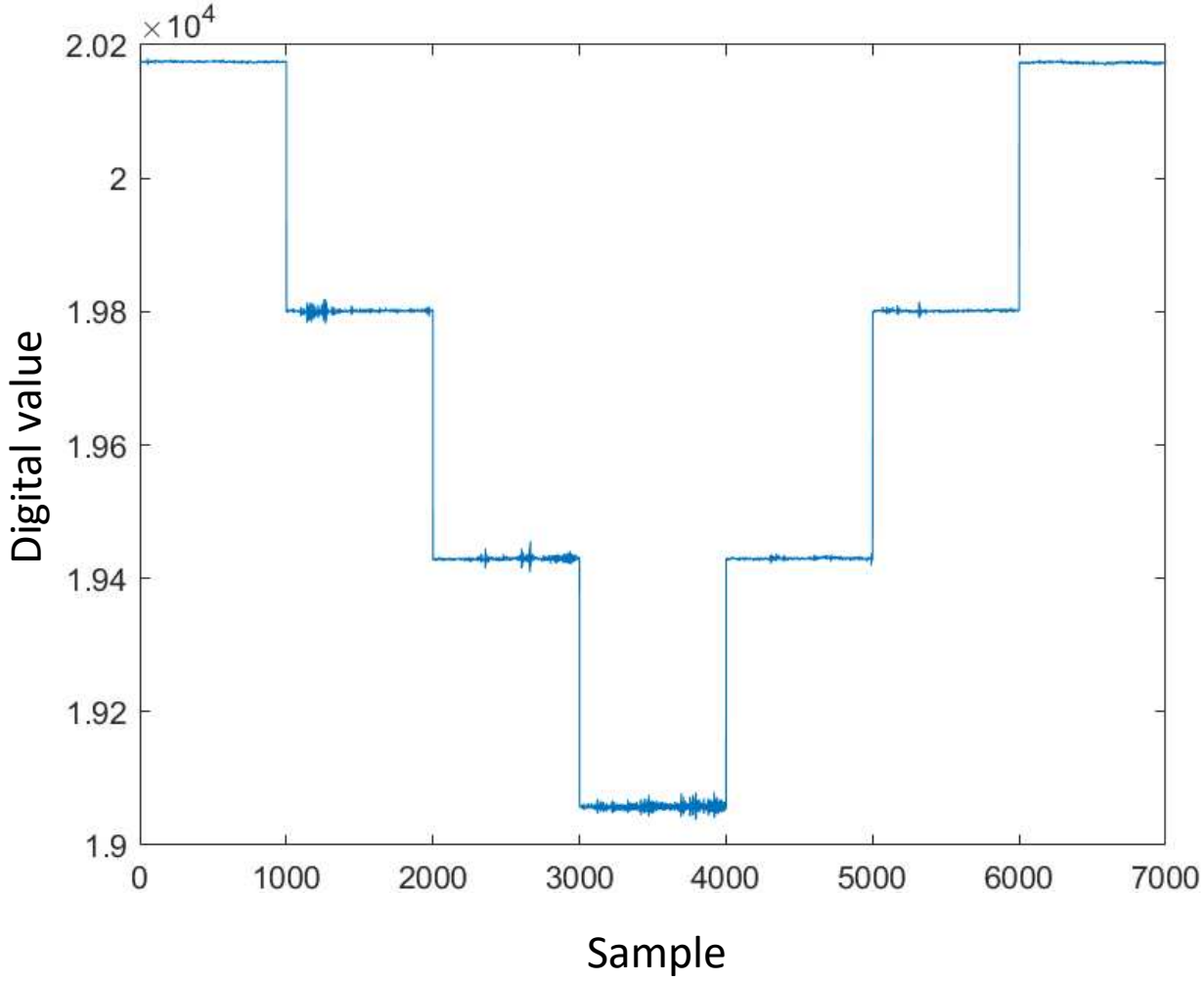
```
V_average = acquired_values_average; %* gain_value;
figure(1);
grid on;
title('CHARACTERIZATION CURVE OF SINGLE CELL');
plot( force , V_average );
hold on;
plot( force , V_average ,'x' );
xlabel('Force [N]');
ylabel('Digital value');
legend({'CHARACTERIZATION CURVE', 'steps average'},'Location','southeast');
saveas(figure(1),'CHARACTERIZATION_CURVE_CELL','jpg');
```



CODICI: TARATURA CELLE DI CARICO



Caratterizzazione cella 1



Caratterizzazione cella 2

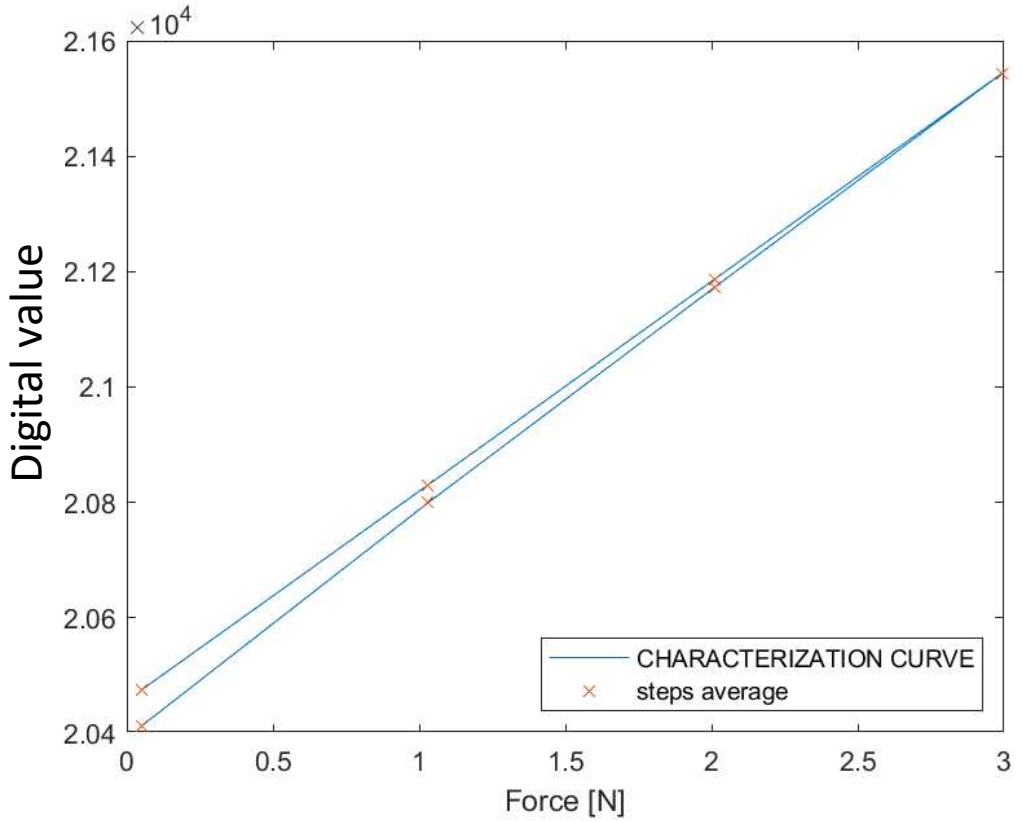
CODICI: TARATURA CELLE DI CARICO

UNIVERSITÀ DEGLI STUDI DI PADOVA

CALCOLO COEFFICIENTI DELLA CARATTERIZZAZIONE

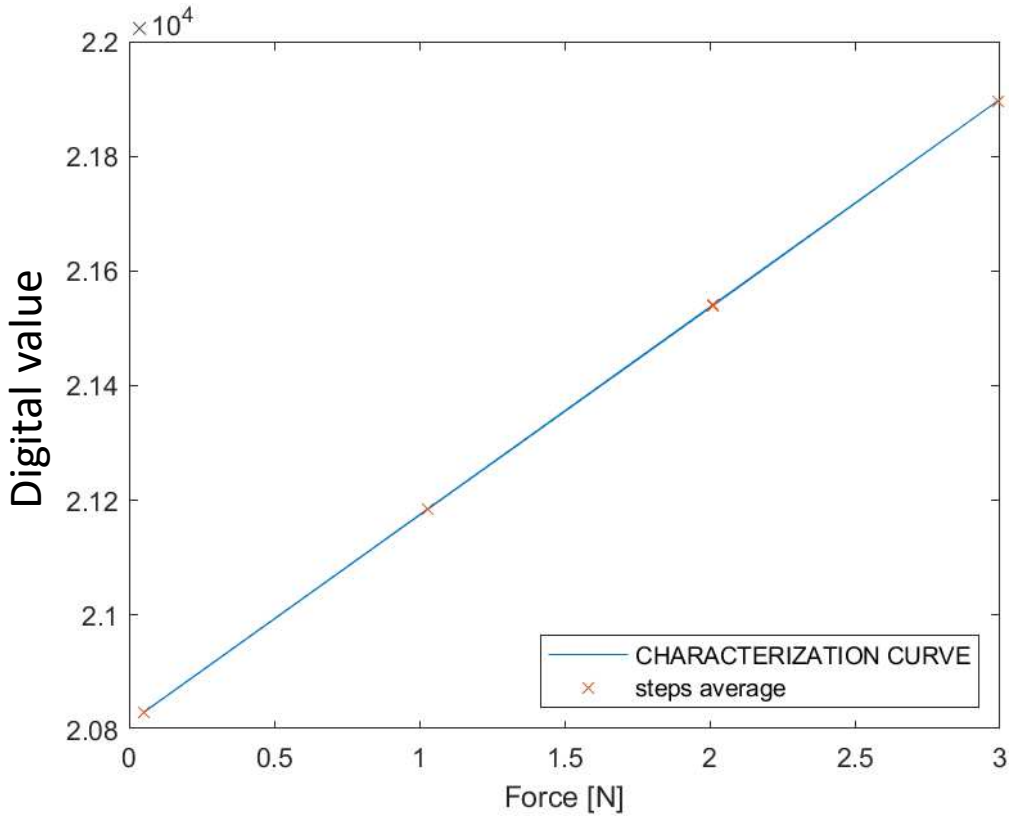
```
%% COEFFICIENTS
K_update(cell_number + 1 ,:) = polyfit( force , V_average , 1 )
K(cell_number + 1 ,:)
flag = input("Press 'y' to save new values or 'n' to discard.");
if flag == 'y'
    save('K_parameters',"K");
end
```

Tramite la funzione polyfit si è in grado di associare il segnale digitalizzato della cella alla forza corrispondente.
La funzione considera i punti di coordinate Forza-Segnale e con il metodo dei minimi quadrati calcola i coefficienti della retta che meglio approssima i punti sul grafico

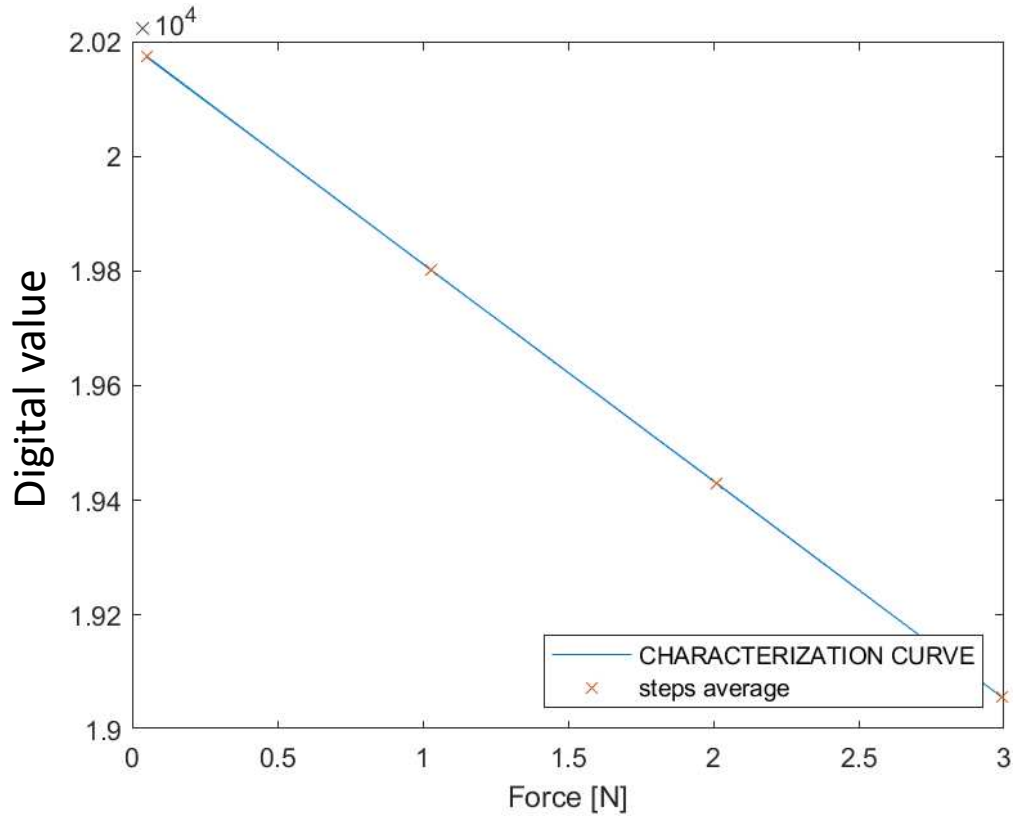


Caratterizzazione cella 0

CODICI: TARATURA CELLE DI CARICO



Caratterizzazione cella 1

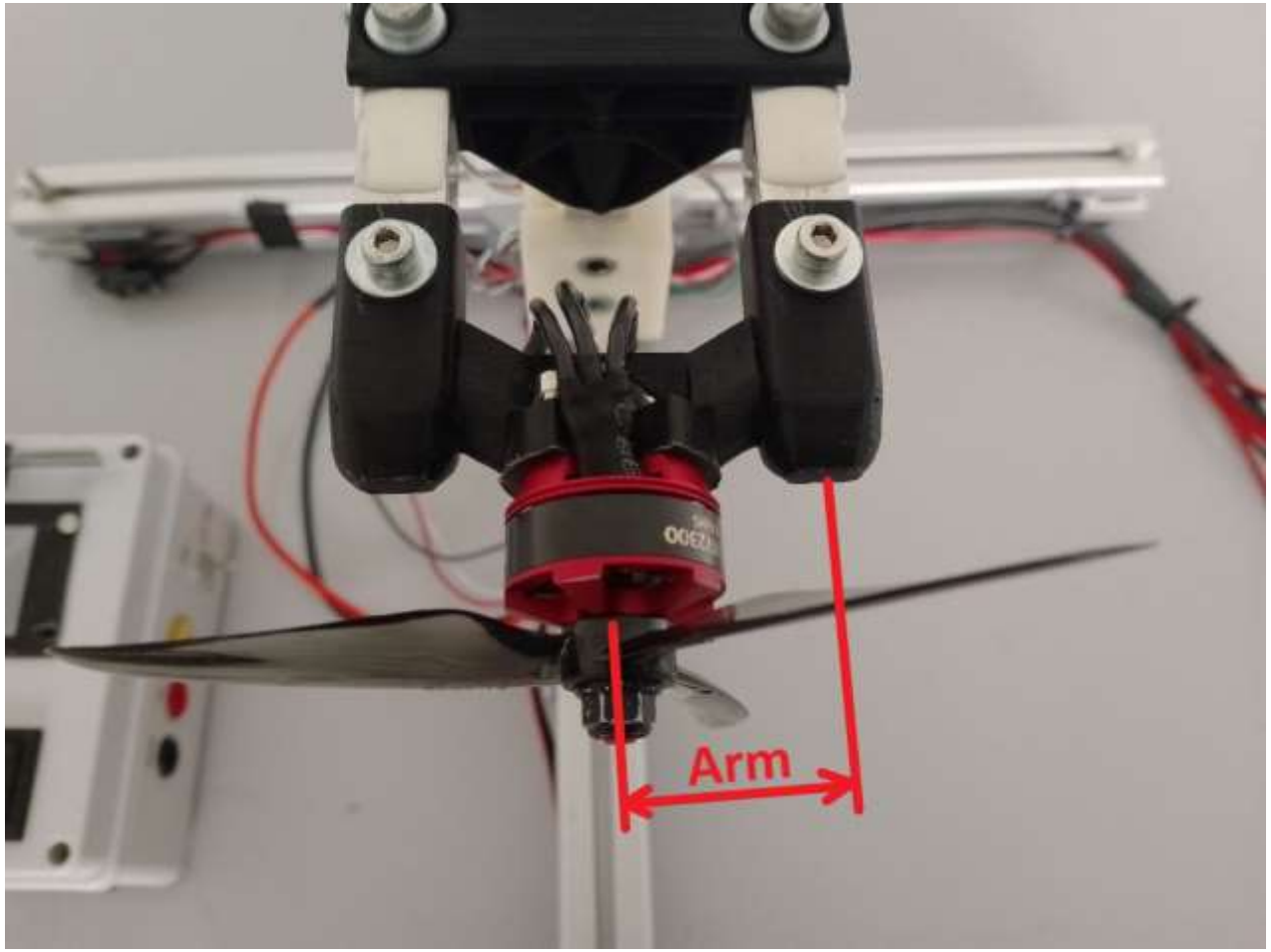


Caratterizzazione cella 2

CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

```
%% PARAMETRI ACQUISIZIONE
load('K_parameters', "K");
K0 = K(1,:);
K1 = K(2,:);
K2 = K(3,:);
arm = 0.028;
sample_quantity = 100
cell_quantity = 3
step_quantity = 9
```



CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

CALIBRAZIONE DELL' ESC

```
disp("ESC initialization...");
read(s,1,'char');%step min
disp("Min. duty value");
read(s,1,'char');%step max
disp("Max. duty value");
read(s,1,'char');%step min
disp("Min. duty value");
read(s,1,'char');%step mid
disp("Mid. duty value");
disp("ESC Ready.");
```

L'Electronic Speed Controller utilizzato necessita della calibrazione prima dell'utilizzo.
In questa procedura si definisce il range del valore di duty cycle utilizzato.

```
bool StopReady = HIGH;
attachInterrupt(STOP_PIN, stop_function, StopReady);

  //MOTOR TEST
if(message=='p'){

  //INIZIALIZZAZIONE ESC
  send_info="t16.txt=\"Test_Motor\"";
  writeString(send_info);

  send_info="t0.txt=\"Init\""; //esc status
  writeString(send_info);


  //int duty_start[]={25,250,25,114};
  for(i=0 ; i<4 ; i++){
    Serial.write('i');
    analogWrite(PWM_PIN,duty_start[i]);

    send_info="n0.val=";
    send_info.concat(duty_start[i]);
    writeString(send_info);

    delay(4500);
  }

  send_info="t0.txt=\"OK\"";
  writeString(send_info);
```

CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

ACQUISIZIONE DATI

```
disp("Data acquisition");
test_values = zeros(step_quantity , sample_quantity , cell_quantity + 1 );
disp("Press to start the test");
pause
write(s, 'm', 'char'); %per far partire l'acquisizione
for i = 1 : step_quantity
    for j = 1 : sample_quantity
        for k = 1 : 4
            if k<=3
                bytes = read(s, 2,'uint8');
                test_values(i,j,k) = typecast(uint8(bytes(1:2)), 'int16');
                if k==1
                    y0(j+(i-1)*sample_quantity)=test_values(i,j,k);
                end
                if k==2
                    y1(j+(i-1)*sample_quantity)=test_values(i,j,k);
                end
                if k==3
                    y2(j+(i-1)*sample_quantity)=test_values(i,j,k);
                end
            end
            if k==4
                bytes = read(s, 4,'uint8');
                test_values(i,j,k) = typecast(uint8(bytes(1:4)), 'single');
                y3(j+(i-1)*sample_quantity)=test_values(i,j,k);
            end
        end
    end
end
disp("Acquisition Done");
```

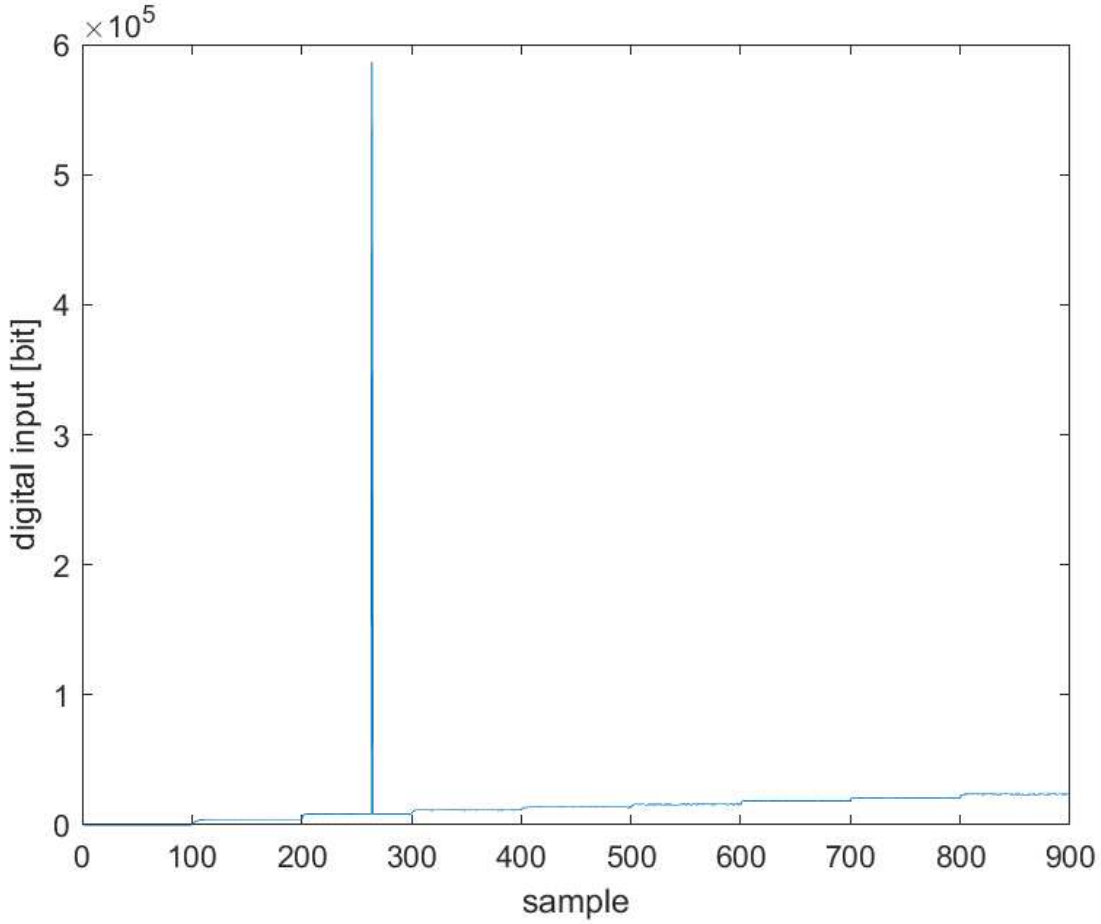
```
while(!Serial.available()){
}
Serial.read();
//DATA EXCHANGE
for(i=0 ; i<9 ; i++){
  analogWrite(PWM_PIN,duty_test[i]);
  send_info="n0.val=";
  send_info.concat(duty_test[i]);
  writeString(send_info);
  for(j=1 ; j<=sample_quantity ; j++){
    for(k=1 ; k<=4 ; k++){// 3 cells + rpm value
      if(k<4){//read from a cell
        int kappa;
        kappa=k-1;
        cell_value = adc.readADC_SingleEnded(kappa);
        byte *pointer=(byte *) &cell_value;
        Serial.write(pointer, 2);
      }
      if(k==4){//read rpm
        beg_2:
        ripetizione();
        if(rpm_acq == false){
          goto beg_2;
        }
        rpm_acq = false;
      }
    }
  }
```

CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

CORREZIONE DATI TELEMETRIA

```
%% CORRECTION FOR SPEED DATA ANOMALIES
disp("Speed data correction.");
while i<=sample_quantity
    while j<=step_quantity
        if test_values(j,i,4)>40000
            test_values(j,i,4) = NaN;
        end
        j=j+1;
    end
    j=1;
    i=i+1;
end
i=1;
j=1;

while i<=step_quantity
    test_values(i,:,4)=fillmissing(test_values(i,:,4) , 'previous');
    i=i+1;
end
i=1;
```



Nelle misurazioni dei giri motore sono spesso presenti errori come in figura dovuti probabilmente all'ESC che non utilizza sensore apposito.
Gli errori si presentano come picchi isolati di velocità molto maggiore di quella a vuoto di poco inferiore a 40000 RPM.

CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

COEFFICIENTI, MEDIA E VARIANZA

```
%% APPLY LINEARIZATION COEFFICIENTS
disp("Linearization");
test_values(:,:,1) = ( test_values(:,:,1)-K0(2) ) / K0(1)
test_values(:,:,2) = ( test_values(:,:,2)-K1(2) ) / K1(1)
test_values(:,:,3) = ( test_values(:,:,3)-K2(2) ) / K2(1)
```

Prima di procedere al calcolo di coppia e spinta, si applicano i coefficienti della caratterizzazione alle misurazioni delle celle di carico e si calcolano media e varianza dei campioni per ogni step.

```
%% AVERAGE OF TEST VALUES
i=1;
j=1;
k=1;
test_values_average = zeros(step_quantity,cell_quantity + 1);
while i<=4
    while j<=step_quantity
        while k<=sample_quantity
            test_values_average(j,i) = test_values_average(j,i)+ test_values(j,k,i);
            k=k+1;
        end
        k=1;
        j=j+1;

    end
    j=1;
    i=i+1;
end
test_values_average;
disp("Calculated averages");
test_values_average = test_values_average / sample_quantity
```

```
%% VARIANCE OF TEST VALUES
i=1;
j=1;
test_values_variance = zeros(step_quantity,cell_quantity + 1);
while i<=4
    while j<=step_quantity
            test_values_variance(j,i)=(sum((test_values(j,:,i)-test_values(j,i)).^2))/(sample_quantity);
            k=k+1;
        j=j+1;
    end
    j=1;
    i=i+1;
end
```

CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

CALCOLO DELLA COPPIA E DELLA SPINTA

```
%% TORQUE ,FORCE AND SPEED CALCS

torque_cell_1 = zeros(step_quantity,2);
torque_cell_2 = zeros(step_quantity,2);
force_cell_0 = zeros(step_quantity,2);
angular_velocity_average = zeros(step_quantity,2);

torque_cell_1 = [ ( (test_values_average(:,2) - test_values_average(1,2)) * arm) , test_values_variance(:,2) * (arm^2) ]  ;
torque_cell_2 = [ ( (test_values_average(:,3) - test_values_average(1,3)) * arm) , test_values_variance(:,3) * (arm^2) ]  ;

torque_average = zeros(step_quantity,2);

force_cell_0(:,1) = abs( test_values_average(:,1) - test_values_average(1,1) ) ;
force_cell_0(:,2) = test_values_variance(:,1);

i=1;
while i<=step_quantity
    torque_average(i,1) = (abs( torque_cell_1(i,1) ) + abs( torque_cell_2(i,1) ));
    torque_average(i,2) = ( torque_cell_1( i , 2 ) + torque_cell_2( i , 2 ) );

    angular_velocity_average(i,1) = ( ( test_values_average(i,4) ) * 2 * pi ) / 60;

    i=i+1;
end
```

Dai valori medi di forza misurata si ricavano la coppia del motore calcolando quelle delle singole celle orizzontali, si sommano poi i contributi. Inoltre i dati dei giri motore, espressi in RPM, vengono convertiti in radianti al secondo.

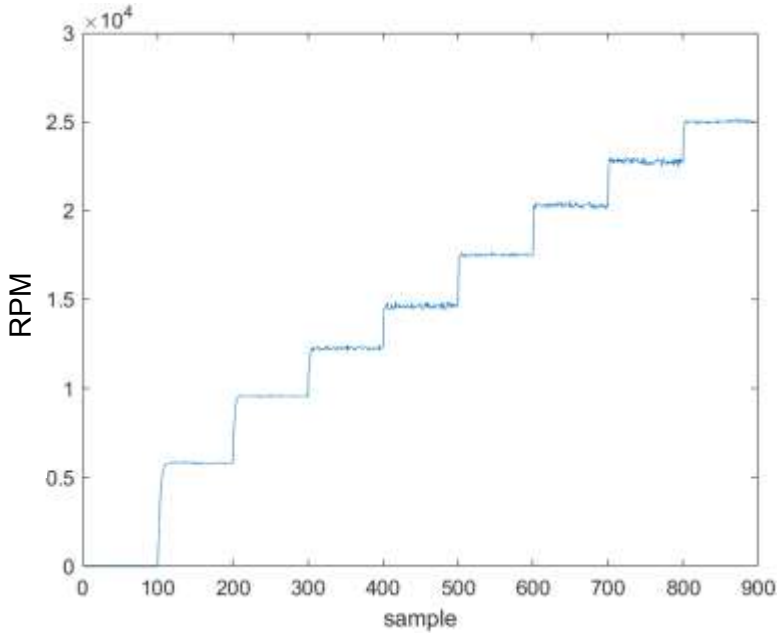
CODICI: ACQUISIZIONE ED ELABORAZIONE DATI

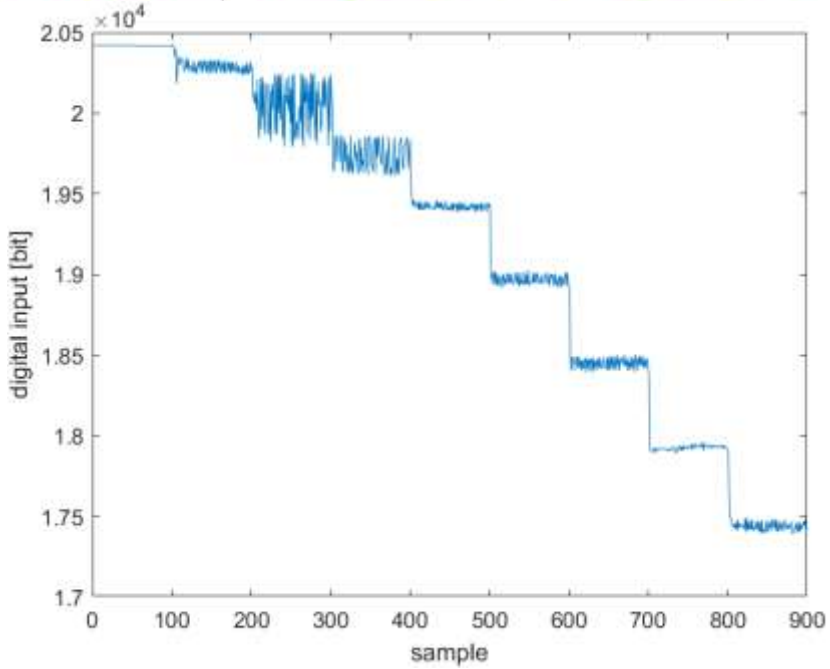
UNIVERSITÀ DEGLI STUDI DI PADOVA

GRAFICI E CALCOLO COEFFICIENTI

```
%% PLOTS  (INCLUDED VARIANCE IN PLOTS)

%force cell 0
figure(1);
plot(angular_velocity_average(:,1) , force_cell_0(:,1));
xlabel('ANGULAR VELOCITY [rad/s]');
ylabel('THRUST [N]');
hold on;

figure(2);
plot(angular_velocity_average(:,1) , torque_average(:,1));
xlabel('ANGULAR VELOCITY [rad/s]');
ylabel('TORQUE [N*m]');

figure(3);
plot(y0);
xlabel('sample');
ylabel('digital input [bit]');
figure(4);
plot(y1);
xlabel('sample');
ylabel('digital input [bit]');
figure(5);
plot(y2);
xlabel('sample');
ylabel('digital input [bit]');
figure(6);
plot(y3);
xlabel('sample');
ylabel('RPM');
```

```
%% COEFFICIENTS CALC. AND SAVE DATA
torque_cell_1
torque_cell_2

CF = polyfit( angular_velocity_average(:,1).^2 , force_cell_0(:,1) , 1 );
CT = polyfit( angular_velocity_average(:,1).^2 , torque_average(:,1) , 1 );

thrust_coefficient = CF(1)
torque_coefficient = CT(1)

test_number = input('Insert the test number : ');
save('Workspace_acquisizione_spinta_tre_celle');
filename = sprintf( '%s_%d' , 'Risultati_prova' , test_number );
save(filename, "force_cell_0", "torque_average", "angular_velocity_average", "thrust_coefficient", "torque_coefficient");
```





CODICI: ANALISI DATI

UNIVERSITÀ DEGLI STUDI DI PADOVA

DEFINIZIONE VARIABILI E CARICAMENTO PROVE

```
%% PARAMETERS
step_quantity = 9;
test_quantity = input('Number of tests analyzed:');

force_results=[];
torque_results=[];
RPM_results=[];
thrust_coefficient_result=[];
torque_coefficient_result=[];

force_average_CF = zeros(9,1);
torque_average_CP = zeros(9,1);
error_average_force = zeros(9,1);
error_average_torque = zeros(9,1);
```

Vengono definite le variabili in cui salvare i risultati delle prove precedenti e quelli, considerando più prove, con lo scopo di ottenere risultati più affidabili.

```
%% LOADING DATA OF PREVIOUS TESTS

for i=1:test_quantity
    %loading data
    filename = sprintf('%s_%d','Risultati_prova',i);
    %force
    load(filename, 'force_cell_0');
    force_results = vertcat( force_results , force_cell_0 );
    %torque
    load(filename, 'torque_average');
    torque_results = vertcat(torque_results,torque_average);
    %angular velocity
    load(filename, 'angular_velocity_average');
    RPM_results = vertcat( RPM_results , angular_velocity_average );
    %coefficients
    load(filename, 'thrust_coefficient');
    thrust_coefficient_result = vertcat( thrust_coefficient_result , thrust_coefficient );
    load(filename, 'torque_coefficient');
    torque_coefficient_result = vertcat( torque_coefficient_result , torque_coefficient );
    %reorder data
    force_results(:,1) = abs(force_results(:,1));
    force_results = sortrows(force_results,1);
    torque_results = sortrows(torque_results,1);
    RPM_results = sortrows(RPM_results,1);

end
```

CODICI: ANALISI DATI

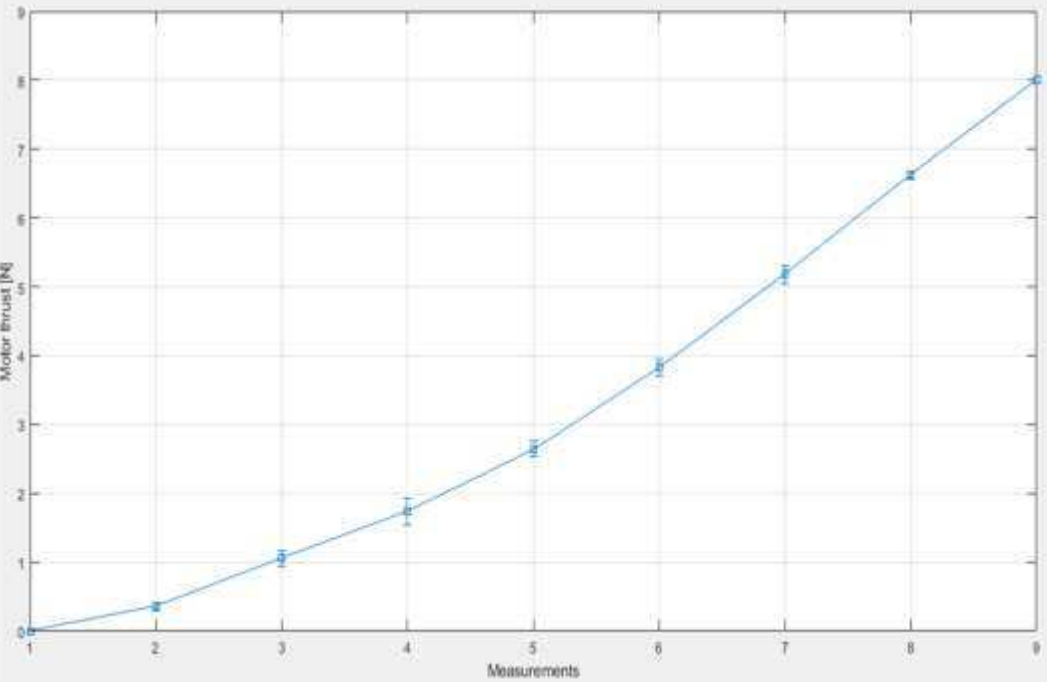
UNIVERSITÀ DEGLI STUDI DI PADOVA

CALCOLO RISULTATI E GRAFICI

```
%% FINAL VALUES

numero_prove = size(torque_results, 1)/step_quantity;
thrust_coefficient_average = sum( thrust_coefficient_result(:,1)) / numero_prove;
torque_coefficient_average = sum( torque_coefficient_result(:,1)) / numero_prove;
j=0;

for i=1:step_quantity
    dati_duty = force_results((j*numero_prove+1):((j+1)*numero_prove),:);
    final_values(i,1)= sum(dati_duty(:,1)) / numero_prove;
    final_values(i,2) = sum(dati_duty(:,2)) / (numero_prove^2);

    dati_duty = torque_results((j*numero_prove+1):(i*numero_prove),:);
    final_values(i,3)= sum(dati_duty(:,1)) / numero_prove;
    final_values(i,4) = sum(dati_duty(:,2)) / (numero_prove^2);

    dati_duty = RPM_results((j*numero_prove+1):(i*numero_prove),:);
    final_values(i,5)= sum(dati_duty(:,1)) / numero_prove;

    %verifica coefficienti di coppia e forza
    force_average_CF(i) = -(final_values(i,5).^2) * thrust_coefficient_average;
    torque_average_CP(i) = (final_values(i,5).^2) * torque_coefficient_average;

    error_average_force(i)=abs( force_average_CF(i) - final_values(i,1) );
    error_average_torque(i)=abs( torque_average_CP(i) - final_values(i,3) );

    j=j+1;

end

save('Workspace_analisi_risultati');
```

```
%%  PLOTS
figure(3);
x = final_values(:,5).^2;
y = final_values(:,1);
plot(x, y,'r');
hold on;
plot(x, force_average_CF, 'b');
grid on;
legend('Average force from cell', 'Average force from coefficients');
xlabel('Velocity^2 [rad^2/s^2]');
ylabel('Motor thrust [N]');

%FORCE & VARIANCE
figure(4);
errorbar(final_values(:,1),error_average_force(:,1), '-s');
grid on;
xlabel('Measurements');
ylabel('Motor thrust [N]');

%TORQUE & SPEED
figure(5);
x = final_values(:,5).^2;
y = final_values(:,3);
plot(x, y, 'r');
hold on;
plot(x, torque_average_CP, 'b');
grid on;
legend('Average torque from cell', 'average torque from coefficients');
xlabel('Velocity^2 [rad^2/s^2]');
ylabel('Motor torque [N*m]');

%TORQUE & VARIANCE
figure(6);
errorbar(final_values(:,3),error_average_torque(:,1), '-s');
grid on;
xlabel('Measurements');
ylabel('Motor torque [N*m]');
```

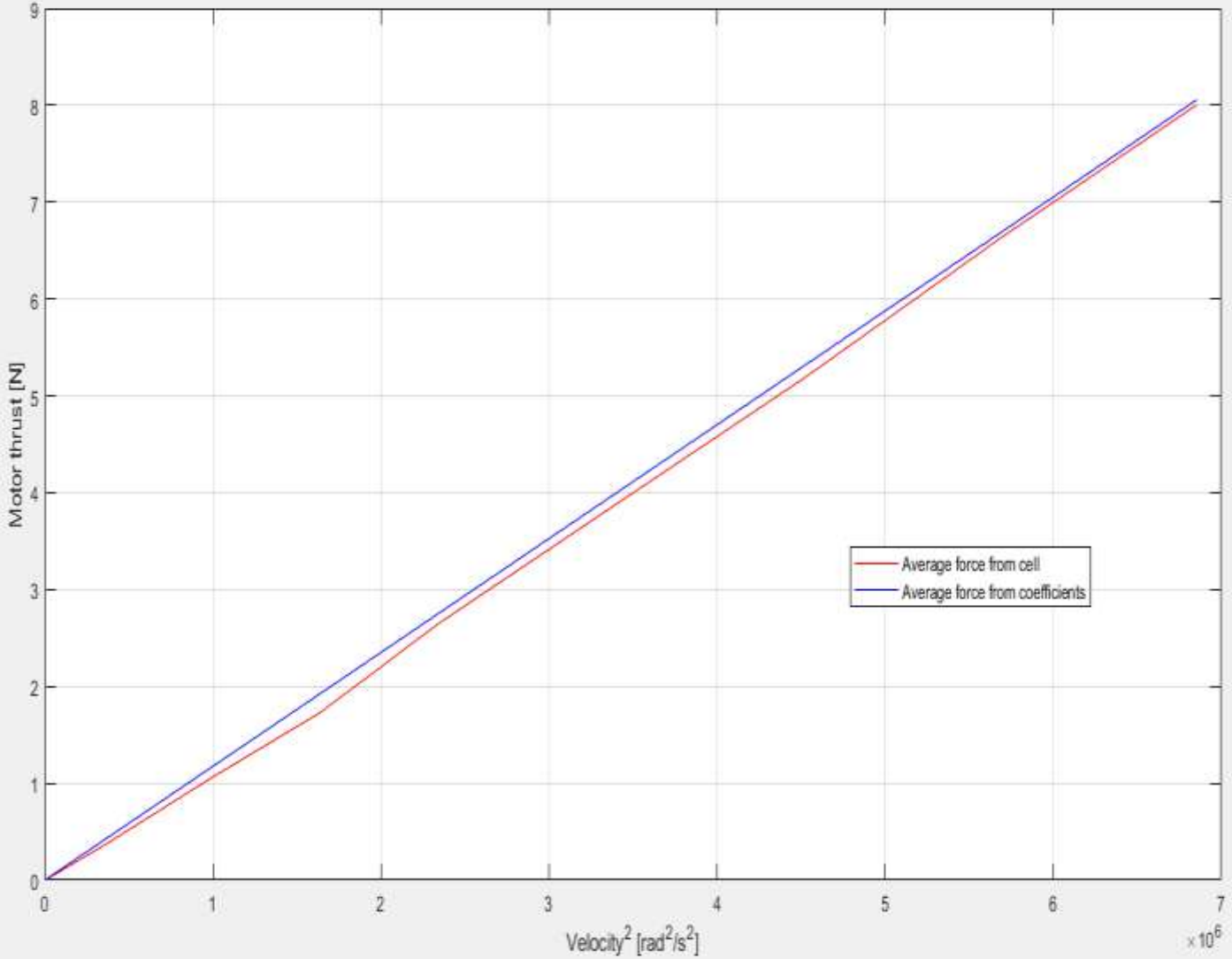
CODICI: GRAFICI E RISULTATI



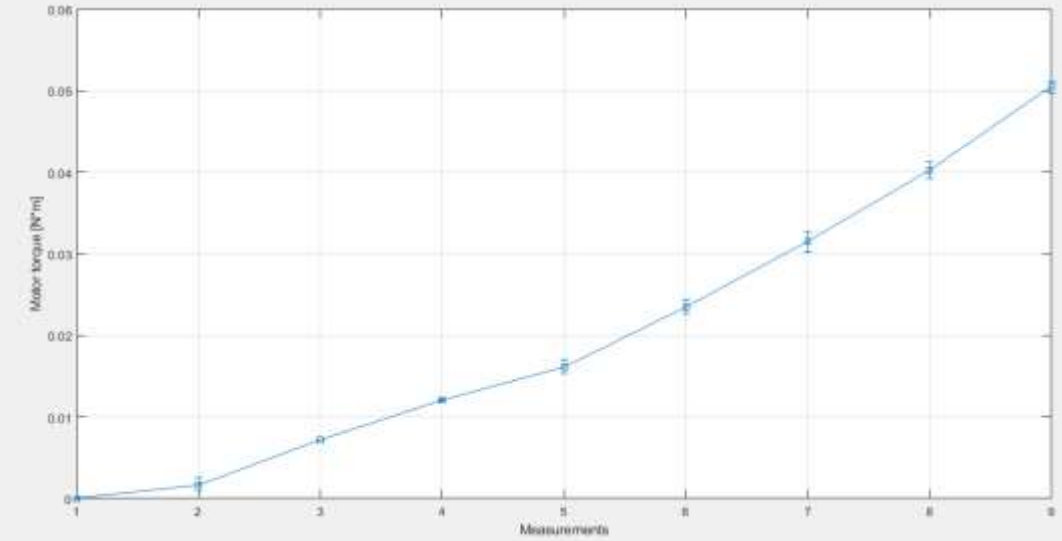
Aumento della spinta massima misurata dell'azionamento a parità di tensione e corrente rispetto alla configurazione di partenza.

S_{max} : 6N ⟶ 8N

Valore confermato confrontando risultati di prove effettuate con azionamenti ed eliche simili a parità di tensione e corrente



CODICI: GRAFICI E RISULTATI



Kv = 2300 [RPM/V]

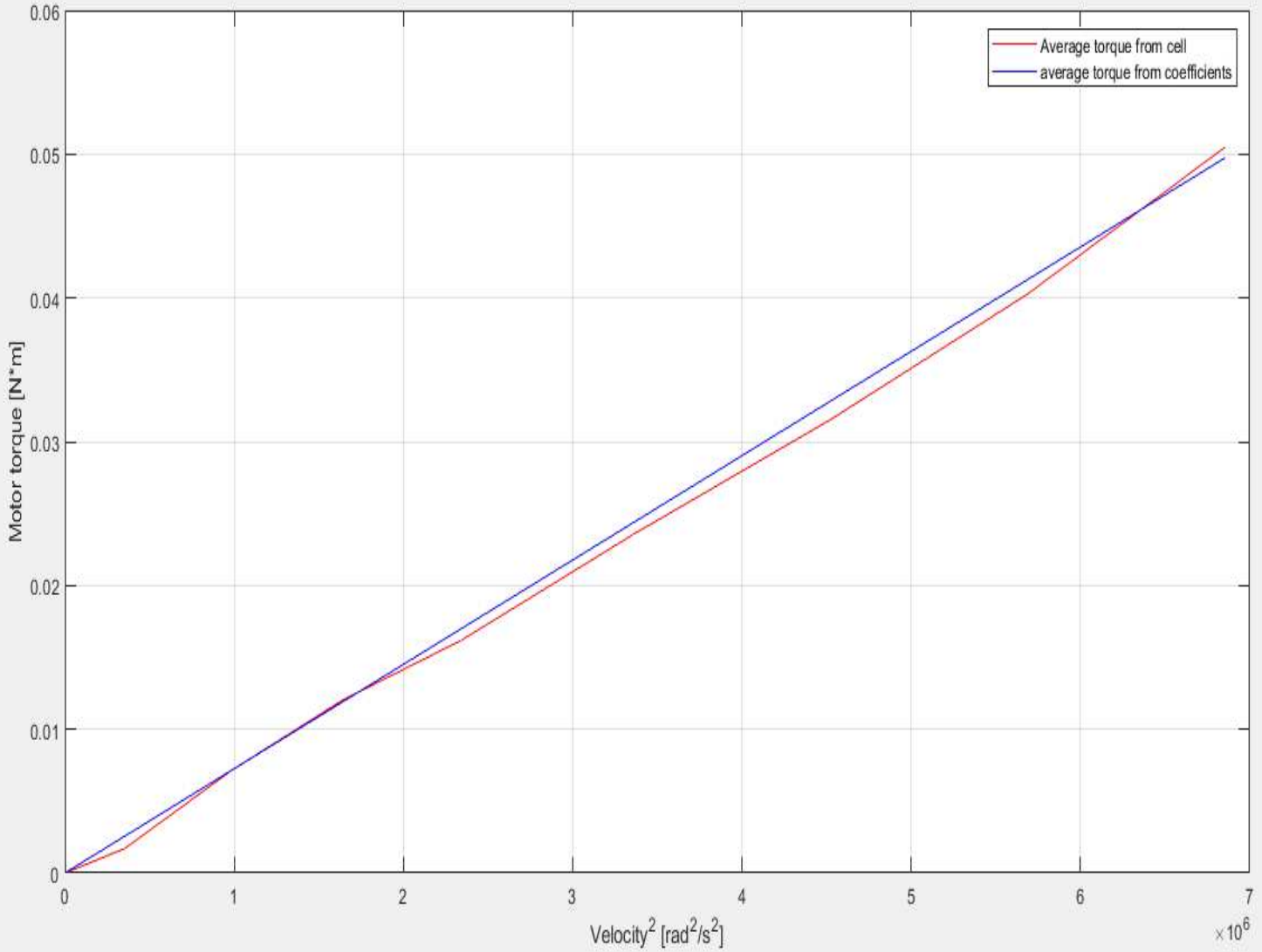
Wo = Kv Vn = 2300 * 16 = 36800 RPM

Kv = 240,85 [rad/s*V] Kt = 1/Kv = 0,00415 Nm/A

Tmax = Kt * In = 0,00415 * 17 = 0,0705 Nm

Tm,max = 0,06 Nm Tm,prec = 0,015 Nm

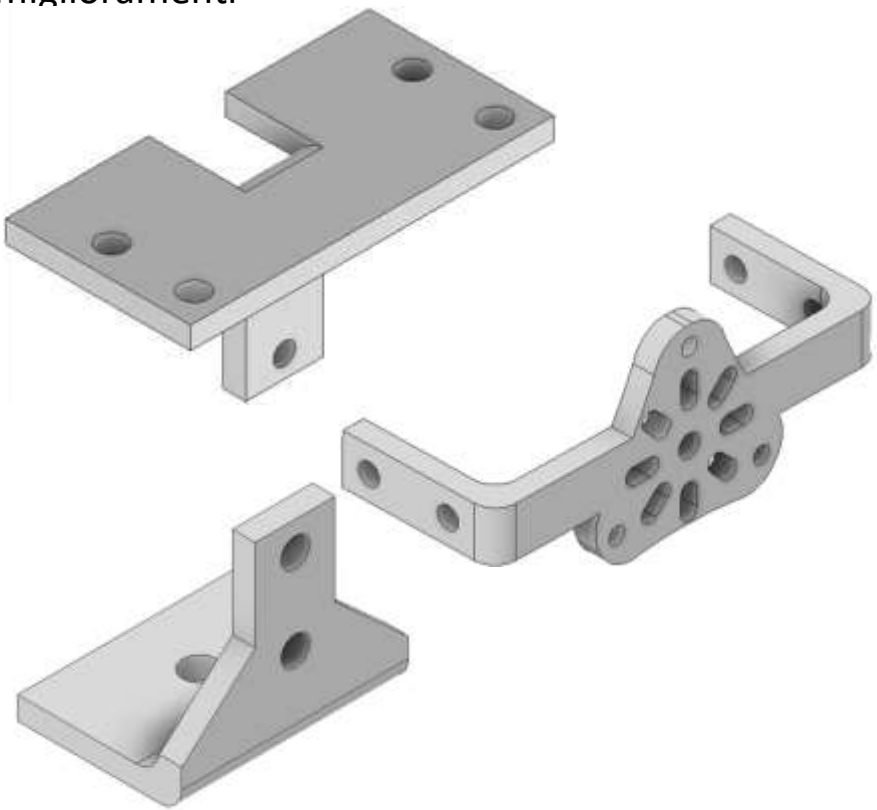
Errore rispetto alla coppia teorica di circa il 16%

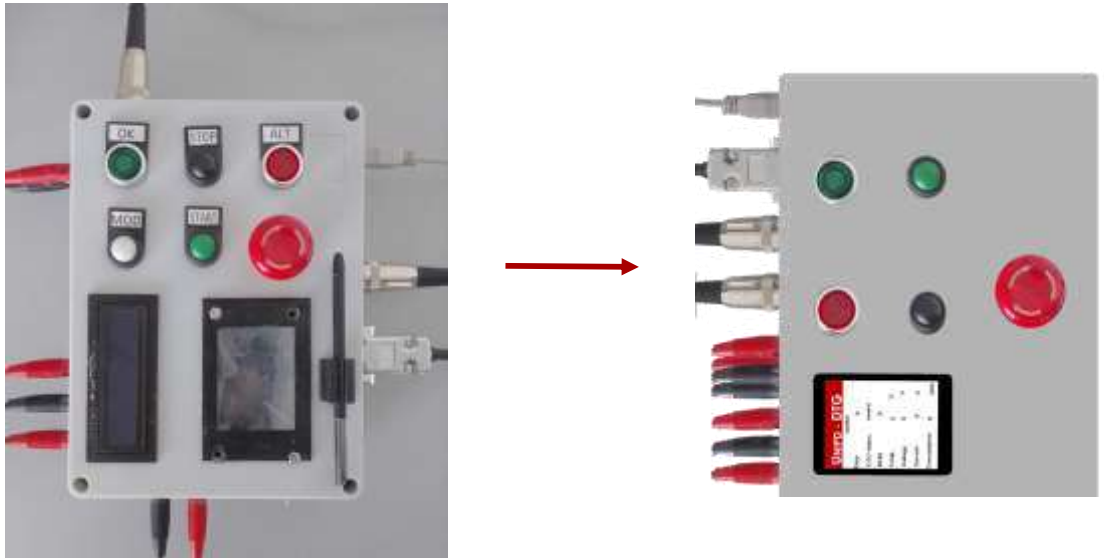


SVILUPPI FUTURI

Dopo le svariate prove e modifiche fatte al banco prova sono stati pensati alcuni possibili miglioramenti che riguardano la qualità dei risultati e l'affidabilità del sistema.
Tra i miglioramenti più importanti troviamo la:

- Realizzazione supporti in metallo per le celle di carico
- Modifica del layout del quadro elettrico
- Modifica della procedura di caratterizzazione delle celle
- Implementazione di sonda di temperatura e sensore di velocità esterno connessi ad arduino





SI RINGRAZIANO

GIULIA MICHIELETTO
STEFANO MICHIELETTO
MASSIMILIANO BERTONI
ANDREA PETUCCO
ROBERTO LOSCO

UNIVERSITÀ
DEGLI STUDI
DI PADOVA