MATCALCULATION::

Grenseverdi=XX;

Avvik(k) = Lyd(k)- nullpunkt; %nullpunkt er første Lydmåling.

if Avvik(k) > Grenseverdi && Flanke(k)==0 %Avvik fra nullpunkt er større enn

Flanke(k)=1; % en manuell satt grenseverdi

Tid(k)=toc; %Tid ved registrert klapp

Ts = Tid(k)- Tid(k-1); %Tida mellom hvert klapp

if Ts<1; %tidskritte mindre en 1 sek, starter diskre tellervar.

n=1+n;

elseif Ts>1; %tidskritt mer en 1 sek, tidskritt nulstilles

n=1;

end

elseif Avvik(k) < Grenseverdi %resette flankeverdi når det ikke er lyd

Flanke(k)=0;

else

Flanke(k)= 2+1; %andre verdier over grenseverdien vil bli

%forskjellig fra 1

end

MOTORPOWER:::

if online

if n==1

Power(B)=2; %Robot kjører forover ved 1 klapp

Power(C)=2;

elseif n==2

Power(B)=0; %Robot stopper ved 2 klapp

Power(C)=0;

elseif n==3

Power(B)=-2; %Robbot kjører babover ved 3 klapp.

Power(C)=-2;

end

motorC.Power = Power(C) ;

motorC.SendToNXT();

motorB.Power = Power(B);

motorB.SendToNXT();

% nullp(k) = nullpunkt; %Updates zero variable used for plotting

% nulll(k) = 0; %Updates actual zero variable for plotting

% LydFiltrert(k) = FIR\_filter(Lyd(1:k),m); %Calls FIR function for k

% Ts(k-1) = Tid(k)-Tid(k-1); %Calculates the time from last timestamp to current

avvik(k) = Lyd(k)-nullpunkt; %Calculates the deviation

% filtrertavvik(k) = LydFiltrert(k) - nullpunkt;

% LydIntegrert(k) = EulerForover(Lyd(k-1),avvik(k-1),Ts(k-1)); %Calls Euler function

% LydFiltrertIIR(k) = IIRfilter(Lyd(k),LydFiltrert(k-1)); %Calls IIR function

% LydIntegrertT(k) = Trapes(LydIntegrert(k-1),avvik(k-1:k),Ts(k-1)); %Calls Trapes function

% avvikderivert(k-1) = Derivasjon(avvik(k-1:k),Ts(k-1));

% LydDerivert(k-1) = Derivasjon(LydFiltrert(k-1:k), Ts(k-1)); %Calls Derivasjon function

if Lyd(k) > grenseverdi

n=1;

i=1;

AntallKlapp=1;

tic

Tid(n) = 0;

while Tid(n) < 5

n=n+1;

Tid(n)= toc;

PowerB=0;

PowerC=0;

if avvik(k) > grenseverdi && Flanke==0

i=i+1;

AntallKlapp(1:i)=1 ;

Flanke = 1 ;

elseif avvik(k) < grenseverdi && Flanke ==1

Flanke =0 ;

end

end

else

Flanke=0;

end

% nullp(k) = nullpunkt; %Updates zero variable used for plotting

% nulll(k) = 0; %Updates actual zero variable for plotting

% LydFiltrert(k) = FIR\_filter(Lyd(1:k),m); %Calls FIR function for k

% Ts(k-1) = Tid(k)-Tid(k-1); %Calculates the time from last timestamp to current

% avvik(k) = Lyd(k)-nullpunkt; %Calculates the deviation

% filtrertavvik(k) = LydFiltrert(k) - nullpunkt;

% LydIntegrert(k) = EulerForover(Lyd(k-1),avvik(k-1),Ts(k-1)); %Calls Euler function

% LydFiltrertIIR(k) = IIRfilter(Lyd(k),LydFiltrert(k-1)); %Calls IIR function

% LydIntegrertT(k) = Trapes(LydIntegrert(k-1),avvik(k-1:k),Ts(k-1)); %Calls Trapes function

% avvikderivert(k-1) = Derivasjon(avvik(k-1:k),Ts(k-1));

% LydDerivert(k-1) = Derivasjon(LydFiltrert(k-1:k), Ts(k-1)); %Calls Derivasjon function

if Lyd(k) > grenseverdi

n=1;

i=1;

AntallKlapp=1;

tic

Tid(n)= toc;

while Tid(n) < 5

n=n+1;

Tid(n)= toc;

Sound(n) = GetSound(SENSOR\_2);

PowerB=0;

PowerC=0;

avvik(n) = Sound(n)- nullpunkt;

if Sound(n) > grenseverdi && Flanke==0

i=i+1;

AntallKlapp(1:i)=1 ;

Flanke = 1 ;

elseif Sound(n) < grenseverdi && Flanke ==1

Flanke =0 ;

end

end

else

Flanke=0;

end

a

% nullp(k) = nullpunkt; %Updates zero variable used for plotting

% nulll(k) = 0; %Updates actual zero variable for plotting

% LydFiltrert(k) = FIR\_filter(Lyd(1:k),m); %Calls FIR function for k

% Ts(k-1) = Tid(k)-Tid(k-1); %Calculates the time from last timestamp to current

% avvik(k) = Lyd(k)-nullpunkt; %Calculates the deviation

% filtrertavvik(k) = LydFiltrert(k) - nullpunkt;

% LydIntegrert(k) = EulerForover(Lyd(k-1),avvik(k-1),Ts(k-1)); %Calls Euler function

% LydFiltrertIIR(k) = IIRfilter(Lyd(k),LydFiltrert(k-1)); %Calls IIR function

% LydIntegrertT(k) = Trapes(LydIntegrert(k-1),avvik(k-1:k),Ts(k-1)); %Calls Trapes function

% avvikderivert(k-1) = Derivasjon(avvik(k-1:k),Ts(k-1));

% LydDerivert(k-1) = Derivasjon(LydFiltrert(k-1:k), Ts(k-1)); %Calls Derivasjon function

if Lyd(k) > grenseverdi

n=1;

i=1;

AntallKlapp(1,1)=1;

tic

Time(n)= toc;

while Time(n) < 5

n=n+1;

Time(n)= toc;

Sound(n) = GetSound(SENSOR\_2);

PowerB=0;

PowerC=0;

if Sound(n) > grenseverdi && Flanke==0

i=i+1;

AntallKlapp(1:i)=1 ;

Flanke = 1 ;

elseif Sound(n) < grenseverdi && Flanke ==1

Flanke =0 ;

end

end

else

x=length(AntallKlapp);

end

% nullp(k) = nullpunkt; %Updates zero variable used for plotting

% nulll(k) = 0; %Updates actual zero variable for plotting

% LydFiltrert(k) = FIR\_filter(Lyd(1:k),m); %Calls FIR function for k

% Ts(k-1) = Tid(k)-Tid(k-1); %Calculates the time from last timestamp to current

% avvik(k) = Lyd(k)-nullpunkt; %Calculates the deviation

% filtrertavvik(k) = LydFiltrert(k) - nullpunkt;

% LydIntegrert(k) = EulerForover(Lyd(k-1),avvik(k-1),Ts(k-1)); %Calls Euler function

% LydFiltrertIIR(k) = IIRfilter(Lyd(k),LydFiltrert(k-1)); %Calls IIR function

% LydIntegrertT(k) = Trapes(LydIntegrert(k-1),avvik(k-1:k),Ts(k-1)); %Calls Trapes function

% avvikderivert(k-1) = Derivasjon(avvik(k-1:k),Ts(k-1));

% LydDerivert(k-1) = Derivasjon(LydFiltrert(k-1:k), Ts(k-1)); %Calls Derivasjon function

Fungerer når har debugg modus på

if Lyd(k) > grenseverdi

i=1;

AntallKlapp=1;

tic

Tid(k)= 0;

PowerB=0;

PowerC=0;

while Tid(k) < 5

P04\_GetNewMeasuremeny

if Lyd(k) > grenseverdi && Flanke==0

i=i+1;

AntallKlapp(1:i)=1 ;

Flanke = 1 ;

elseif Lyd(k) < grenseverdi && Flanke ==1

Flanke =0 ;

end

end

else

x=length(AntallKlapp);

end