MATLAB Project
Simulation of the visual environment-photodiode interaction using photoreceptors detection and biomimetism functions

GACHET Théo, GRELLAT-SEVILLA Inès

Ecole Nationale Supérieure des Mines de Saint-Etienne

December 16, 2022

1 The Project

1.1 Context

The project is based on a fly's vision and behavior, through biomimetism. The general purpose is to create a bio-robot imitating the patterns and movements of a fly. Our specific approach focuses on the interaction between the photoreceptors of the fly's eye and the perceived environment.

The point of the project is for the robot to move between two walls composed of stripes of different shades of gray. To achieve this project, we will assume that our fly only moves in a single direction and we want to simulate it vision considering the Gaussian sensitivity of the photodiodes.

For simplicity's sake, we are only going to consider two photoreceptors facing the same wall, and we are going to try to program the photoreceptors' response.

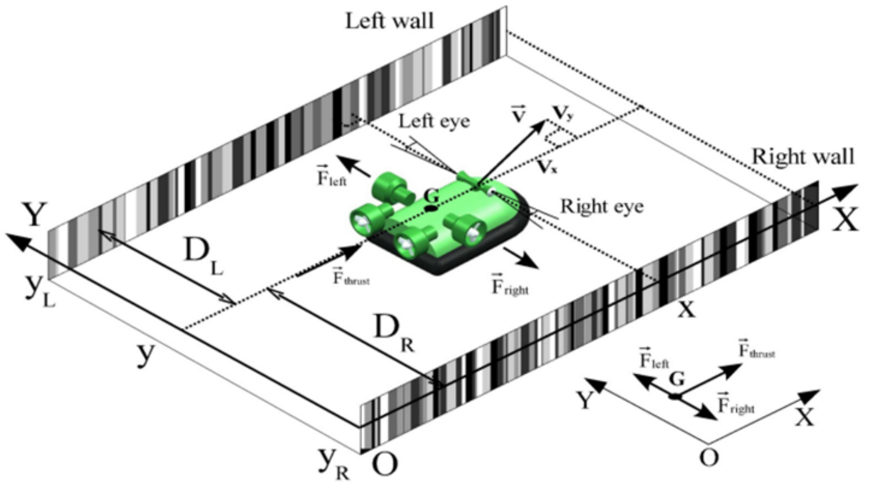


Figure 1: Global description of the project

1.2 Segmentation

Our project is composed of five major big steps:

1. Create the environment function names env().
2. Create a matrix to avoid recalculating the integrals of the Gauss curve.
3. Create the function giving the color observed by the fly according to time.
4. Analyze the fly's movment with the photoreceptors simulated in Simulink.
5. Put the output through a bandpass filter and finally convert it into 16 bits.

Project's planning: September : step 1 / October : step 2 / November : steps 3 & 4 / December : step 5

The photoreceptor has a gaussian-shaped sensitivity, meaning that what is in the robot's peripheral vision won't be as crucial in the calculus as what's right in front of it.

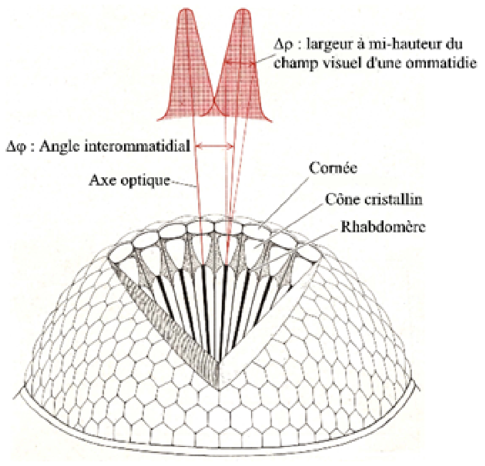

Δρ : largeur à mi-hauteur du champ visuel d'une ommatidie
Δφ : Angle interommatidial
Axe optique
Cornée
Cône cristallin
Rhabdomère

Figure 2: Photoreceptors

The look-up table contains the integral of the gaussian function which represents the area of a certain shade of gray seen by the photoreceptor.

The environment is a four meters long gray-striped wall, with distinct shades. The distinction between the shades will be the benchmarks in our calculus.

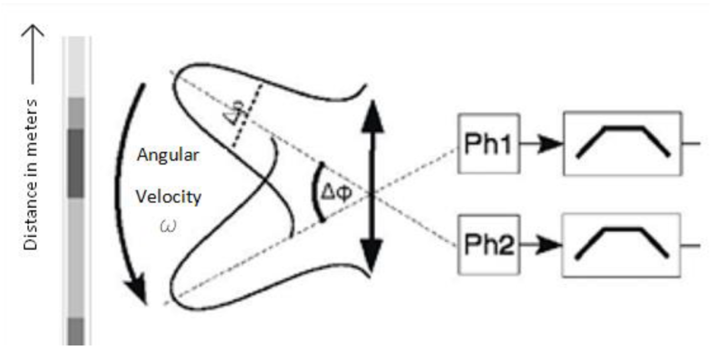


Figure 3: Sensibility of the two photoreceptors which are staggered by an angle $\Delta\phi$

2 Environment

The first step in our biomimetism project is to model the environment in which our fly will evolve. The model will only contain one dimension (the x-axis) for a 4-meter distance.

The key element in our model is that the environment is randomly generated by grey bands which length is included between 10mm and 100mm. The gray shade of every stripe is determined randomly, as well as the width of thereof. Finally, the environment will have a resolution of δx = 1mm.

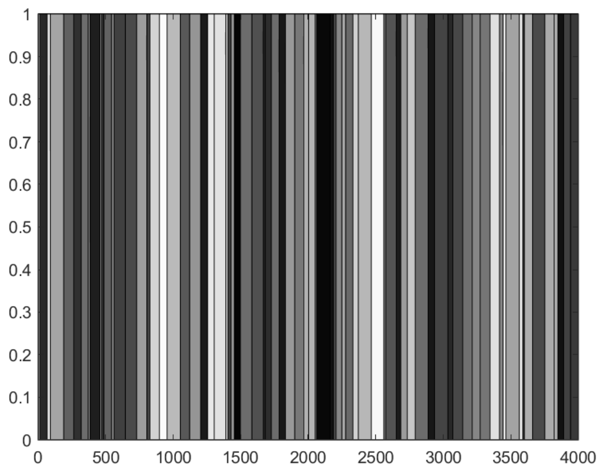


Figure 4: The randomly generated environment

Our code uses a matrix with two rows: one for the width and one for the color. We randomly generated those two values and it stops when it reaches the total length of four meters. If the last stripe is a bit too long, we cut it so it can fit into the four meters, without losing the random aspect of the environment.

3 Look-up tables

We just saw what the environment will look like, but our fly won't see it as clearly as we do. Indeed, it is the separation between two stripes that will determine the angle used later to calculate the sensitivity of the photoreceptor for the stripe considered. We will use a matrix to get what the photoreceptor perceives.

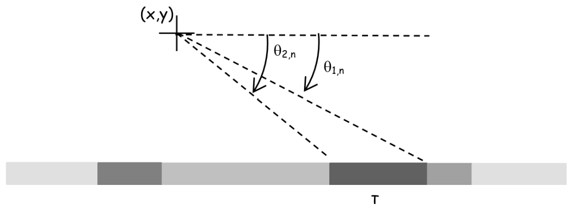


Figure 5: Setting of the angles subtended by a contrast

Thus, the next step consists in creating a matrix with all the predetermined areas needed later in the program, the A matrix. It will be a computational tool we will use to simplify the integral of the Gaussian curve of photoreceptors.

Basically, this matrix is used as a computational tool to simplify the integral of the Gaussian curve of photoreceptors as it represents a correspondence table thus allowing us to estimate the integral of the Gaussian sensitivity between two contrast transition angles. This matrix A will considerably reduce the execution time while modeling the photodiode outputs Ph1 and Ph2. Note that our matrix will be pre-calculated with a resolution $\delta\phi = 0.005$.

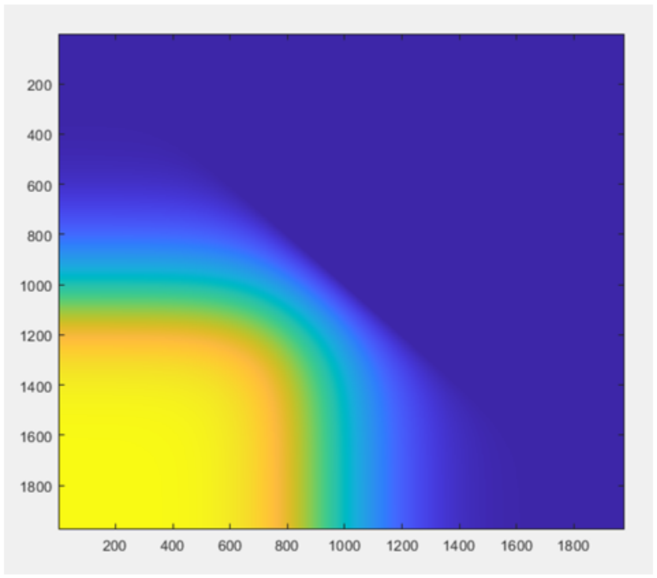


Figure 6: The matrix A is a lower triangular matrix of size 1976x1976 ($2.6 * \Delta\rho = 1976$)

This function's parameters are the $\Delta\rho$ angle and the angular step (i.e. the resolution). At the end, the call A(angle1, angle2) returns the integral between two angular positions. Indeed, the photoreceptor output is calculated by piecewise integrating various luminances present in its visual field and weighting them by the Gaussian angular sensitivity.

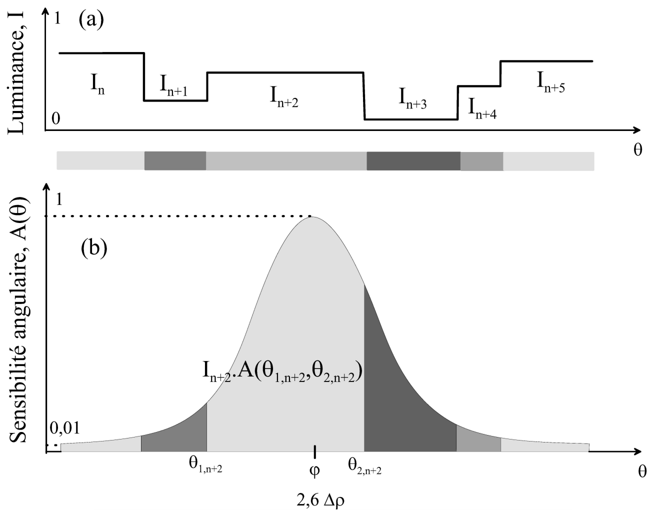


Figure 7: The output depends of the luminance and the Gaussian angular sensitivity

The sensitivity of our photoreceptors depending on the angle is calculated with the Gaussian function, with ϕ the azimuthal position and σ the standard deviation.

$$f(\phi) = e^{-\frac{(\phi-\phi_{max})^2}{2\sigma^2}}$$

$$\text{where} \begin{cases} \phi \text{ is the azimuthal position in the visual field of the photoreceptor} \\ \phi_{max} \text{ is the azimuthal position giving maximum photoreceptor response} \\ \sigma \text{ is the width at 95\% of the height of the Gaussian (standard deviation)} \\ \Delta\rho \text{ is the width at 50\% of the height of the Gaussian (angle of acceptance)} \end{cases}$$

We also know that a relation exists between $\Delta\rho$ and σ :

$$\Delta\rho = 2\sqrt{2ln(2)}\sigma$$

At the end, the matrix contains every precalculated area of this gaussian function, allowing us to get exactly how much the photoreceptor is sensitive to a given stripe.

4 Simulating the photodetectors

We successfully created an environment for our experiment. Now, we have to deduce from it the way the fly's photoreceptors will perceive the size and color of every band. The process we used is complex and can be explained by the organigram below.

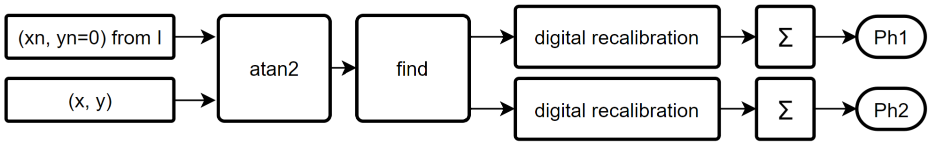


Figure 8: Organigram explaining the process used to obtain the photoreceptors output

4.1 Visual field of the photodetectors

The MatLab function "find" allowed us to select the angles that are in the visual field of each of the photoreceptors, knowing that they are separated by $\pm\frac{\Delta\phi}{2}$ from the wall.

$$\text{for Ph1,} \; -90° - \frac{\Delta\phi}{2} - 1.3\Delta\rho \leq \theta_n \leq -90° - \frac{\Delta\phi}{2} + 1.3\Delta\rho$$

$$\text{for Ph2,} \; -90° + \frac{\Delta\phi}{2} - 1.3\Delta\rho \leq \theta_n \leq -90° + \frac{\Delta\phi}{2} + 1.3\Delta\rho$$

4.2 Digital recalibration

During this step, we just want to make sure that $1 \leq \theta_n \leq 1976$ in order to then exploit the matrix A.

$$recalibration = \frac{angle - angle_{min}}{step}$$

4.3 The matrix A

The matrix A is thus used to determine the output of each of the photoreceptors by sommation:

$$Ph = \int_0^{2\pi} I(\theta).A(\theta).\,d\theta \approx \sum_{n=1}^{N} I_n.A(\theta_{1,n}, \theta_{2,n})$$

Please note that while calculating the global perception of the photoreceptor, we use a discrete sum instead of an integral to simplify the computer's calculating process.

5 Results and photodetector outputs

5.1 Simulation parameters

In order to correctly visualize the output of the two photodetectors, we used the Simulink software and we assumed that the fly moves parallel to the wall at 30 cm and a constant speed of 0.5 m.s^{-1}, which is the role of the ramp block in our simulation.

Finally, we added a second order digital bandpass filter [20 Hz ; 116 Hz] and converted our data to 16 bits to simulate the analog to digital conversion of the microcontroller, with a temporal sampling frequency of 2 kHz.

5.2 Output visualization

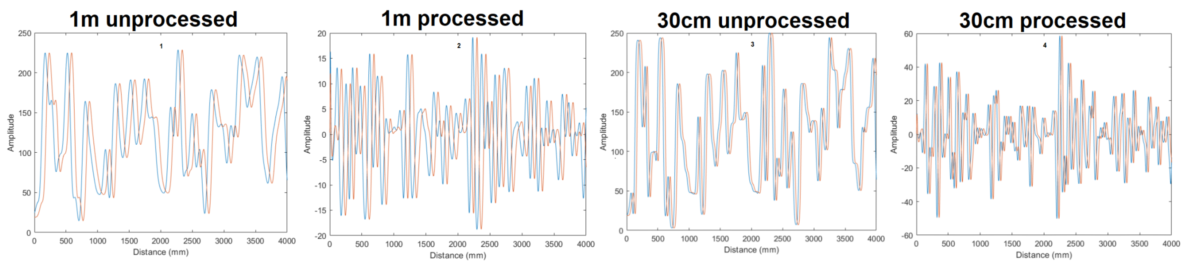


Figure 9: Photodetectors outputs

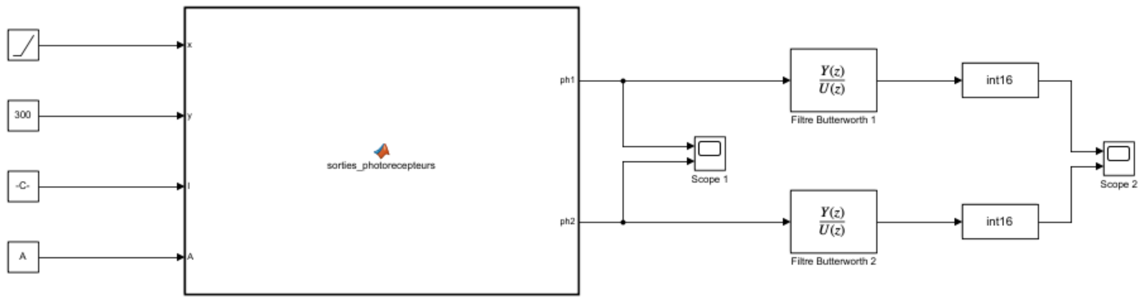


Figure 10: Simulink blocks diagram

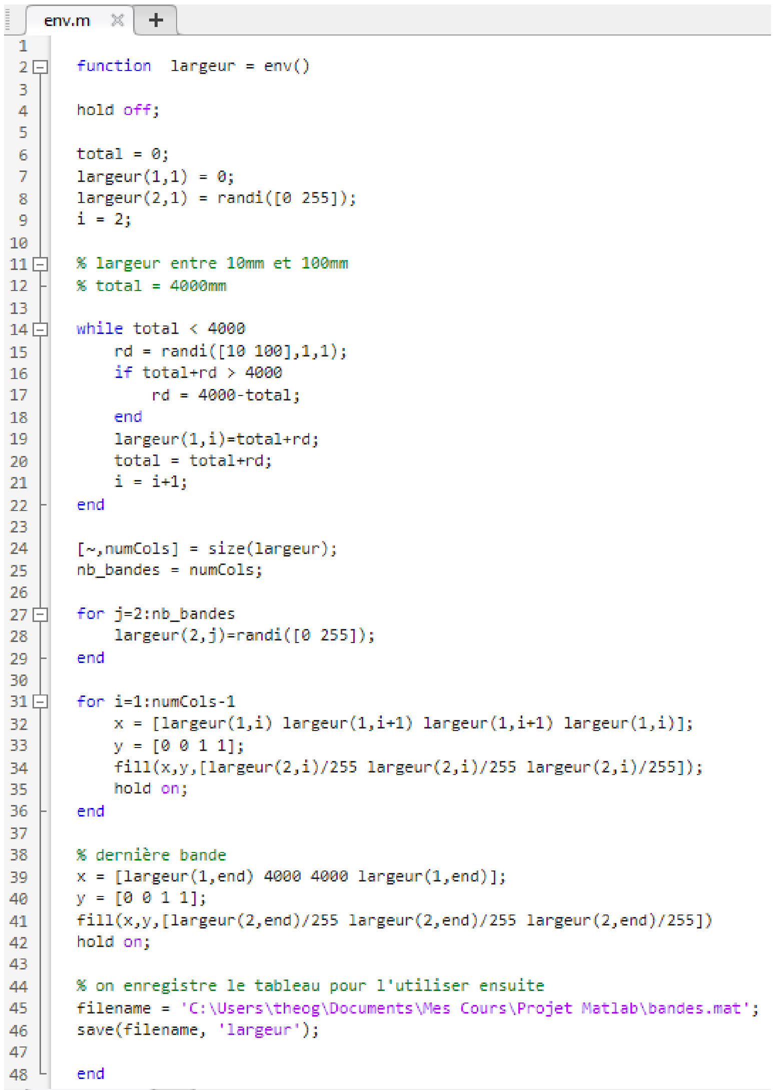
5.3 Interpretation

As we are moving from the wall, the two signals are shifting, it was expectable. Moreover, it is clear that the results for the un-processed output are smoother when we are the farthest from the wall.

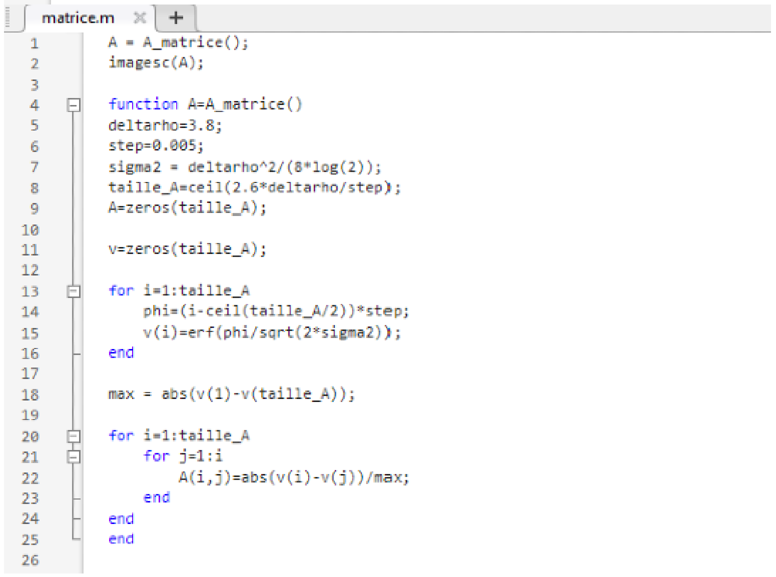
5.4 Conclusion

Our simulated environment is coherent and allows us to obseve the evolution of the photoreceptor sensitivity. Our code is functional and could be implemented on a robot, proving the efficiency of biomimetism combined with new technologies.

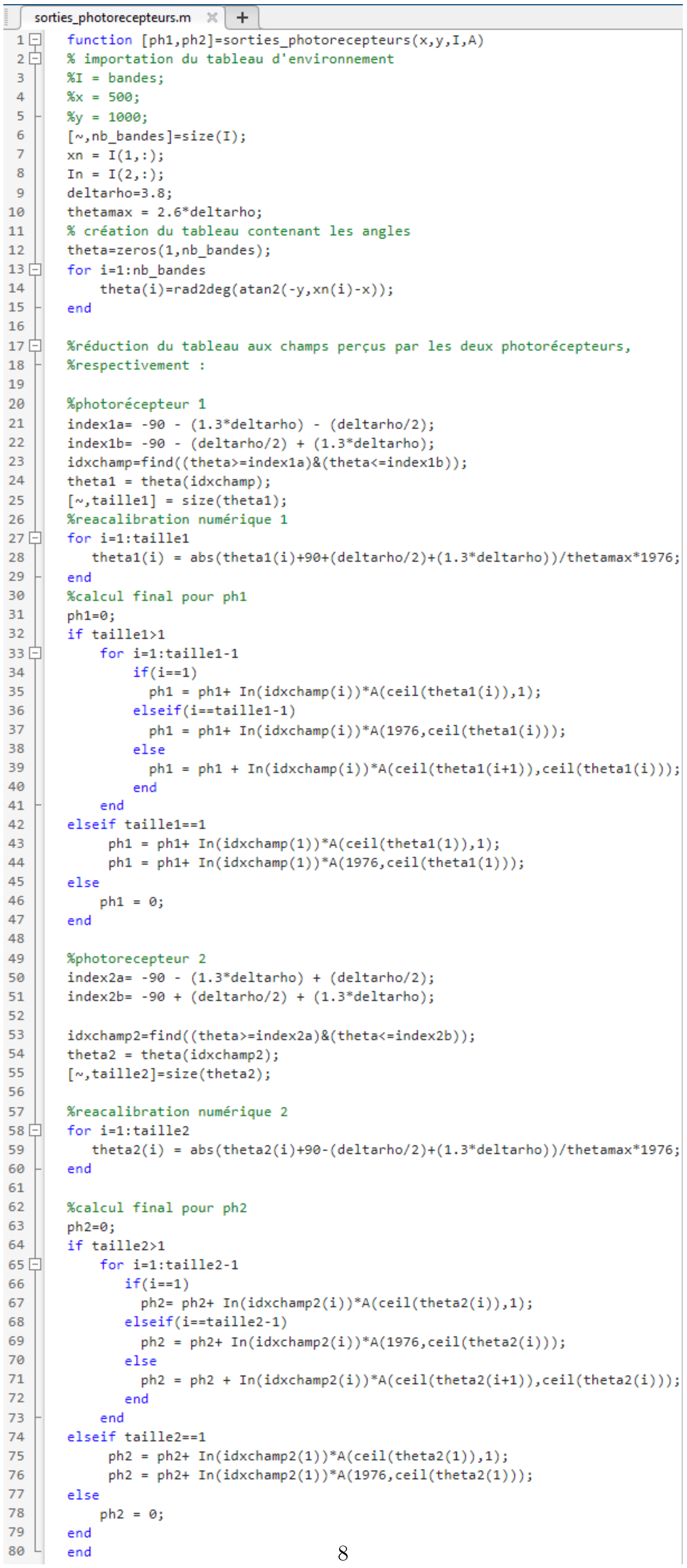
6 Annexes


env.m

```matlab

function  largeur = env()

hold off;

total = 0;
largeur(1,1) = 0;
largeur(2,1) = randi([0 255]);
i = 2;

% largeur entre 10mm et 100mm
% total = 4000mm

while total < 4000
    rd = randi([10 100],1,1);
    if total+rd > 4000
        rd = 4000-total;
    end
    largeur(1,i)=total+rd;
    total = total+rd;
    i = i+1;
end

[~,numCols] = size(largeur);
nb_bandes = numCols;

for j=2:nb_bandes
    largeur(2,j)=randi([0 255]);
end

for i=1:numCols-1
    x = [largeur(1,i) largeur(1,i+1) largeur(1,i+1) largeur(1,i)];
    y = [0 0 1 1];
    fill(x,y,[largeur(2,i)/255 largeur(2,i)/255 largeur(2,i)/255]);
    hold on;
end

% dernière bande
x = [largeur(1,end) 4000 4000 largeur(1,end)];
y = [0 0 1 1];
fill(x,y,[largeur(2,end)/255 largeur(2,end)/255 largeur(2,end)/255])
hold on;

% on enregistre le tableau pour l'utiliser ensuite
filename = 'C:\Users\theog\Documents\Mes Cours\Projet Matlab\bandes.mat';
save(filename, 'largeur');

end
```


matrice.m

```matlab
A = A_matrice();
imagesc(A);

function A=A_matrice()
deltarho=3.8;
step=0.005;
sigma2 = deltarho^2/(8*log(2));
taille_A=ceil(2.6*deltarho/step);
A=zeros(taille_A);

v=zeros(taille_A);

for i=1:taille_A
    phi=(i-ceil(taille_A/2))*step;
    v(i)=erf(phi/sqrt(2*sigma2));
end

max = abs(v(1)-v(taille_A));

for i=1:taille_A
    for j=1:i
        A(i,j)=abs(v(i)-v(j))/max;
    end
end
end

```



```
function [ph1,ph2]=sorties_photorecepteurs(x,y,I,A)
% importation du tableau d'environnement
%I = bandes;
%x = 500;
%y = 1000;
[~,nb_bandes]=size(I);
xn = I(1,:);
In = I(2,:);
deltarho=3.8;
thetamax = 2.6*deltarho;
% création du tableau contenant les angles
theta=zeros(1,nb_bandes);
for i=1:nb_bandes
    theta(i)=rad2deg(atan2(-y,xn(i)-x));
end

%réduction du tableau aux champs perçus par les deux photorécepteurs,
%respectivement :

%photorécepteur 1
index1a= -90 - (1.3*deltarho) - (deltarho/2);
index1b= -90 - (deltarho/2) + (1.3*deltarho);
idxchamp=find((theta>=index1a)&(theta<=index1b));
theta1 = theta(idxchamp);
[~,taille1] = size(theta1);
%reacalibration numérique 1
for i=1:taille1
   theta1(i) = abs(theta1(i)+90+(deltarho/2)+(1.3*deltarho))/thetamax*1976;
end
%calcul final pour ph1
ph1=0;
if taille1>1
    for i=1:taille1-1
        if(i==1)
          ph1 = ph1+ In(idxchamp(i))*A(ceil(theta1(i)),1);
        elseif(i==taille1-1)
          ph1 = ph1+ In(idxchamp(i))*A(1976,ceil(theta1(i)));
        else
          ph1 = ph1 + In(idxchamp(i))*A(ceil(theta1(i+1)),ceil(theta1(i)));
        end
    end
elseif taille1==1
     ph1 = ph1+ In(idxchamp(1))*A(ceil(theta1(1)),1);
     ph1 = ph1+ In(idxchamp(1))*A(1976,ceil(theta1(1)));
else
    ph1 = 0;
end

%photorecepteur 2
index2a= -90 - (1.3*deltarho) + (deltarho/2);
index2b= -90 + (deltarho/2) + (1.3*deltarho);

idxchamp2=find((theta>=index2a)&(theta<=index2b));
theta2 = theta(idxchamp2);
[~,taille2]=size(theta2);

%reacalibration numérique 2
for i=1:taille2
   theta2(i) = abs(theta2(i)+90-(deltarho/2)+(1.3*deltarho))/thetamax*1976;
end

%calcul final pour ph2
ph2=0;
if taille2>1
    for i=1:taille2-1
       if(i==1)
         ph2= ph2+ In(idxchamp2(i))*A(ceil(theta2(i)),1);
       elseif(i==taille2-1)
         ph2 = ph2+ In(idxchamp2(i))*A(1976,ceil(theta2(i)));
       else
         ph2 = ph2 + In(idxchamp2(i))*A(ceil(theta2(i+1)),ceil(theta2(i)));
       end
    end
elseif taille2==1
     ph2 = ph2+ In(idxchamp2(1))*A(ceil(theta2(1)),1);
     ph2 = ph2+ In(idxchamp2(1))*A(1976,ceil(theta2(1)));
else
    ph2 = 0;
end
end
```