c\hough.c

```c
// Objectif : utiliser du multithreading pour améliorer la rapidité
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

int COLONNES;
int LIGNES;
int DISTANCE;

// taille du fichier. update global variables
int* taille(char* filepath){
    FILE* ptr;
    char ch;
    ptr = fopen(filepath, "r");
    if (NULL == ptr) {
        printf("file can't be opened \n");
    }
    int lignes = 0;
    int colonnes = 1;
    int end = 1;
    while (!feof(ptr)) {
        ch = fgetc(ptr);
        if (ch == '\n'){
            lignes ++;
            end = 0;
        }
        else if(ch == ',' && end) {
            colonnes ++;
        }
    }

    LIGNES = lignes;
    COLONNES = colonnes;
    DISTANCE = (int)round(sqrt(COLONNES *COLONNES + LIGNES *LIGNES));
}

```

```c
//retourne la matrice
int* read_img(char* filepath){
    int* matrice = malloc((COLONNES * LIGNES) * sizeof(int));

    FILE* ptr;
    char ch;
    ptr = fopen(filepath, "r");

    if (NULL == ptr) {
        printf("file can't be opened \n");
    }
    int c = 0;
    while (!feof(ptr)) {
        ch = fgetc(ptr);
        if (feof(ptr)){
            fclose(ptr);
            return matrice;
        }
        if (ch == '\n'){
            c++;
        }
        else if(ch == ',') {
            c++;
        }
        else if (ch == ' '){}
        else {
            //printf("%c\n", ch);
            matrice[c] *= 10;

            matrice[c] += (int)ch - 48;
        }
    }
}

void print_matrice(int* mat){
    for (int i = 0; i<LIGNES; i++){
        for (int j = 0; j < COLONNES; j++){
            printf("%d | ", mat[i*COLONNES + j]);
        }
        printf("\n");
    }
```

```
}

double rho(int x, int y, double theta){
    return(x * cos(theta) + y * sin(theta));
}

int* espace_hough(int* image, int precision){
    int* mat_hough = malloc(360 * precision * DISTANCE * sizeof(int));
    int v_rho;
    double theta_rad;
    for (int i = 0; i< 360 * DISTANCE * precision; i++){
        mat_hough[i] = 0;
    }
    for (int y = 0; y < LIGNES; y++){
        for (int x = 0; x < COLONNES; x++){
            if (image[y*COLONNES + x] > 250){
                for (int theta_deg = 0; theta_deg < 360 * precision ;theta_deg ++){
                   theta_rad = theta_deg /(180.0 * precision) * M_PI;
                   v_rho = rho(x, y, theta_rad);
                   if (v_rho > 0){
                        mat_hough[v_rho * 360 * precision + theta_deg] += 1;
                   }
                }
            }
        }
    }
    return(mat_hough);
}


void print_hough(int* mat){
    for (int i = 0; i<360; i++){
        for (int j = 0; j < DISTANCE; j++){
            printf("%d | ", mat[j*360 + i]);
        }
        printf("\n");
    }
}

void save_file(int* mat, int precision){
    FILE *ptr;
```

```
    ptr = fopen("./out.txt","w");

    for (int j = 0; j < DISTANCE; j++){
        for (int i = 0; i<360 * precision; i++){
            fprintf(ptr, "%d", mat[j*360 * precision + i]);
            if (i != 360 * precision - 1){
                fprintf(ptr, "%s",",");
            }
        }
        fprintf(ptr, "%s","\n");
    }

    fclose(ptr);

}


int main()
{
    int precision = 100;
    printf("precision : %d\n", precision);
    char* filepath = "./in.txt";
    taille(filepath);
    printf("Lignes : %d\nColonnes : %d\nnombre de pixels : %d\nDistance : %d\n", LIGNES, COLONNES,(LIGNES)*(COLONNES), DISTANCE);
    int* image = read_img(filepath);
    //print_matrice(image);
    int* espace = espace_hough(image, precision);
    //print_hough(espace);

    save_file(espace, precision);
    return 0;
}
```