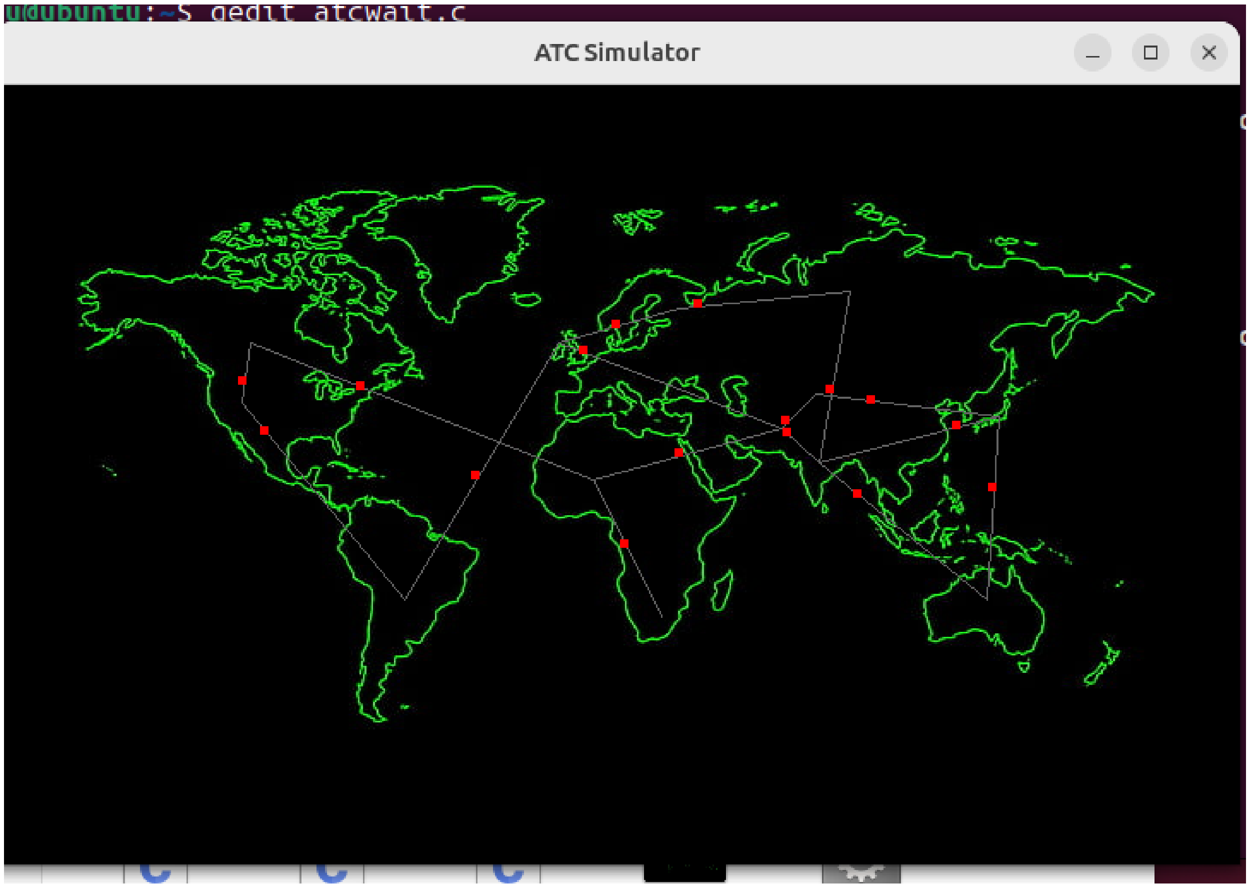
AIR TRAFFIC CONTROL SIM

Air traffic control simulator made using SDL2, semaphores and threads in C language on Ubuntu OS

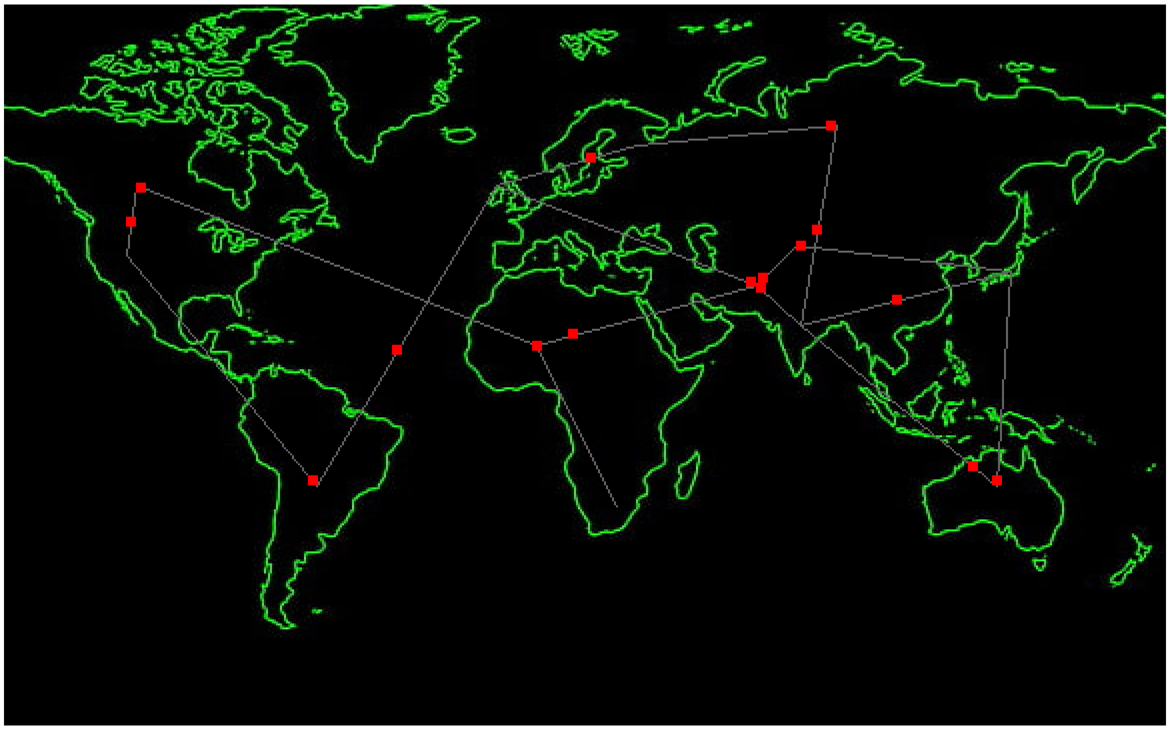


Ibrahim Khalil

21K-3456
BCS-4A

WORKING

There are multiple routes coded into the application on which planes move. The program utilizes multiple threads to display the movements of each aircraft. They are programmed to follow their respective paths and not collide with each other as well.



(In this image, it can be seen that some planes have stopped on the airports to prevent colliding with other planes.)

RESOURCES USED

1. Valve Corporation Simple DirectMedia Layer (SDL2).
2. Ubuntu OS.
3. Threads and semaphores

SOURCE CODE:

```
#include <SDL2/SDL.h>

#include<SDL2/SDL_image.h>

#define SCREEN_WIDTH 728

#define SCREEN_HEIGHT 455


typedef struct {

    float x;

    float y;

} Point;


typedef struct {

    Point start;

    Point end;

} Line;


typedef struct {

    Point position;

    Line path;

    float progress;

} Plane;


void draw_line(SDL_Renderer *renderer, Line line) {
    SDL_RenderDrawLine(renderer, (int)line.start.x, (int)line.start.y, (int)line.end.x,
(int)line.end.y);

}


void draw_plane(SDL_Renderer *renderer, Point position) {

    SDL_Rect rect = { (int)position.x - 2.5, (int)position.y - 2.5, 5, 5 };

    SDL_RenderFillRect(renderer, &rect);

}

void draw_bg(SDL_Renderer *renderer, SDL_Texture* texture){

        SDL_RenderCopy(renderer, texture, NULL, NULL);

}



int main(int argc, char *argv[]) {

SDL_Init(SDL_INIT_VIDEO);


    // Create a window

    SDL_Window *window = SDL_CreateWindow(

        "ATC Simulator",

        SDL_WINDOWPOS_UNDEFINED,

        SDL_WINDOWPOS_UNDEFINED,

        SCREEN_WIDTH,

        SCREEN_HEIGHT,

        SDL_WINDOW_SHOWN
    );


    SDL_Renderer *renderer = SDL_CreateRenderer(window, -1,
SDL_RENDERER_ACCELERATED);



SDL_Surface* surface = IMG_Load("map.jpg");

SDL_Texture* texture = SDL_CreateTextureFromSurface(renderer, surface);

SDL_FreeSurface(surface);





    // Create some planes

    Plane planes[] = {

        //pak-uk

        {

            { 460,200 },

            { { 460, 200 }, { 330,150 } },

            0

        },

        //can--afr

        {

            { 150, 150 },
            { { 150, 150 }, { 350, 230 } },
            0
        },
        //pak-ind
        {
            { 460,200 },
            { { 460, 200 }, { 482,220 } },
            0
        },
        //eur-uk
        {
            { 400,130 },
            { { 400, 130 }, { 330,150 } },
            0
        },
        //rus-ind
        {
            { 500,120 },
            { { 500, 120 }, { 482,220 } },
            0
        },
        //rus-eur
        {
            { 500,160 },
            { { 500, 120 }, { 400,130 } },

            0

        },

        //can--us

        {

            { 150, 150 },

            { { 150, 150 }, { 145, 185 } },

            0

        },

        //china-jap

        {

            { 480,180 },

            { { 480,180 }, { 587, 193 } },

            0

        }

        ,//safr to afr

        {

            { 390, 310 },

            { { 390, 310 }, { 350, 230 } },

            0

        },

        //samr--us

        {

            { 240 , 300 },
            { { 240, 300 }, { 145, 185 } },
            0
        },
        //samr--uk
        {
            { 240 , 300 },
            { { 240, 300 }, { 330, 150 } },
            0
        },
        //china-pak
        {
            { 480,180 },
            { { 480,180 }, { 460, 200 } },
            0
        },
        //pak-afr
        {
            { 460,200 },
            { { 460, 200 }, { 350,230 } },
            0
        },
        //ind - jap
        {
            { 482,220 },
                { { 482, 220 }, { 587,193 } },
                0
            },
            //aus - jap
            {
                { 580,300 },
                { { 580, 300 }, { 587,193 } },
                0
            },
            //aus - ind
            {
                { 580,300 },
                { { 580, 300 }, { 482,220 } },
                0
            }

        };
        int num_planes = sizeof(planes) / sizeof(planes[0]);


        // Start the simulation loop
        Uint32 last_frame_time = SDL_GetTicks();
        while (1) {
            // Handle events
            SDL_Event event;
```

```
while (SDL_PollEvent(&event)) {
    if (event.type == SDL_QUIT) {
        // Quit the program if the window is closed
        SDL_DestroyRenderer(renderer);
        SDL_DestroyWindow(window);
        SDL_Quit();
        return 0;
    }
}

// Calculate the time since the last frame
Uint32 current_time = SDL_GetTicks();
Uint32 elapsed_time = current_time - last_frame_time;
last_frame_time = current_time;

// Clear the screen
SDL_SetRenderDrawColor(renderer, 0, 0, 0, 255);

     draw_bg(renderer,texture);
// Draw the lines
SDL_SetRenderDrawColor(renderer, 100, 100, 100, 100);

for (int i = 0; i < num_planes; i++) {
    draw_line(renderer, planes[i].path);
```

```
        }


    // Update and draw the planes
    SDL_SetRenderDrawColor(renderer, 255, 0, 0, 255);
    for (int i = 0; i < num_planes; i++) {
        // Update the plane's position
        float distance = elapsed_time / 1000.0 * 25;  // 50 pixels per second
        planes[i].progress += distance / SDL_sqrt(SDL_pow(planes[i].path.end.x - 
planes[i].path.start.x, 2) + SDL_pow(planes[i].path.end.y - planes[i].path.start.y, 2));
        if (planes[i].progress > 1) {
            planes[i].progress = 0;
            Point tmp = planes[i].path.start;
            planes[i].path.start = planes[i].path.end;
            planes[i].path.end = tmp;
        }
        planes[i].position.x = planes[i].path.start.x + planes[i].progress * 
(planes[i].path.end.x - planes[i].path.start.x);
        planes[i].position.y = planes[i].path.start.y + planes[i].progress * 
(planes[i].path.end.y - planes[i].path.start.y);


        // Draw the plane
        draw_plane(renderer, planes[i].position);
    }


    // Present the rendered image to the screen
    SDL_RenderPresent(renderer);
}
}
.
```

CONCLUSION

This application can display multiple planes, on different routes and predict collisions and take precautionary measures