Disaster alert system.

2021 Term Project Report



항공우주응용SW1

제출일자 : 2021. 12. 17

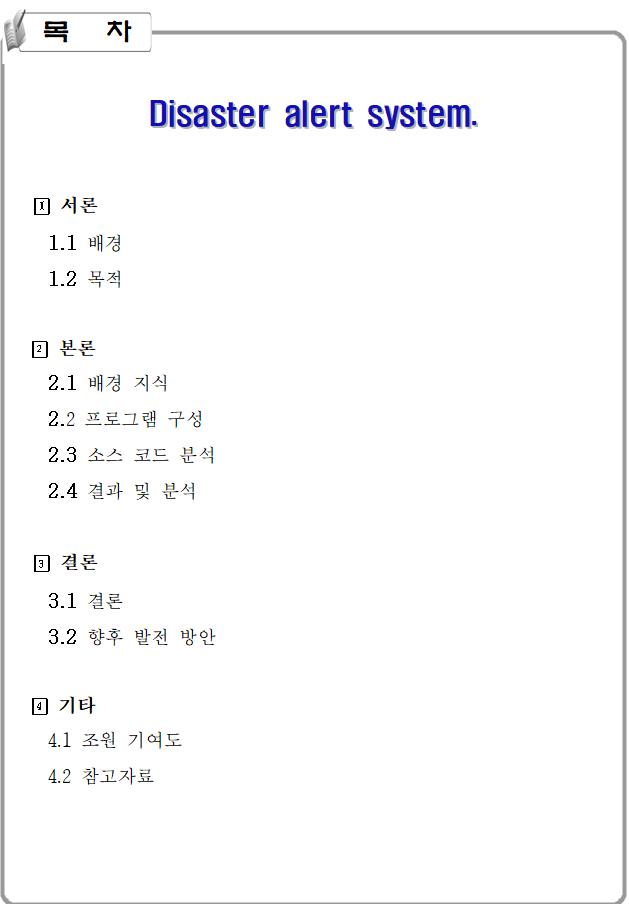
담당 교수 : 박병운

19013177 김동우

19013181 김영섭

19013187 윤태진

18011258 한규동



**서론**

**1.1배경**

일상생활 속에서 우리는 다양한 위급 상황에 직면할 수 있습니다. 갑자기 응급환자가 발생할 수도 있고 갑작스러운 폭우나 폭설로 재난지역에 고립될 수도 있습니다.

하지만 많은 사람들은 재난으로 인한 피해가 자신에게 피해가 가지 않을 거라는 안전 불감증도 많이 가지고 있습니다.

재난경보시스템은 자연적 재난과 일정규모 이상의 사고,폭발,화재 등 사회적(국가적) 재난이 발생할 경우에 경보를 발생합니다. 그렇기에 자신과는 무관하다는 생각을 하지만 재난은 언제나 우리에게 닥쳐올 수 있기에 재난 알림 경보를 조금 더 개인에 초점을 맞추려고 합니다.

**1.2목적**

우리는 태풍이나 지진과 같은 전국적 재난뿐만 아니라 화재나 산사태 같은 국소적 지역에서 일어나는 재난에 대한 알림을 추가하여 사람들에게 전송할 것입니다. 재난이 발생하고 이에 대해 발송되는 문자의 정보를 대폭 추가하여 사람들이 받았을 때 경각심을 가지게 하여 자신의 주변에서 일어난 재난으로 인해 받을 수 있는 2차 피해를 방지할 수 있도록 할 예정입니다.

**본론**

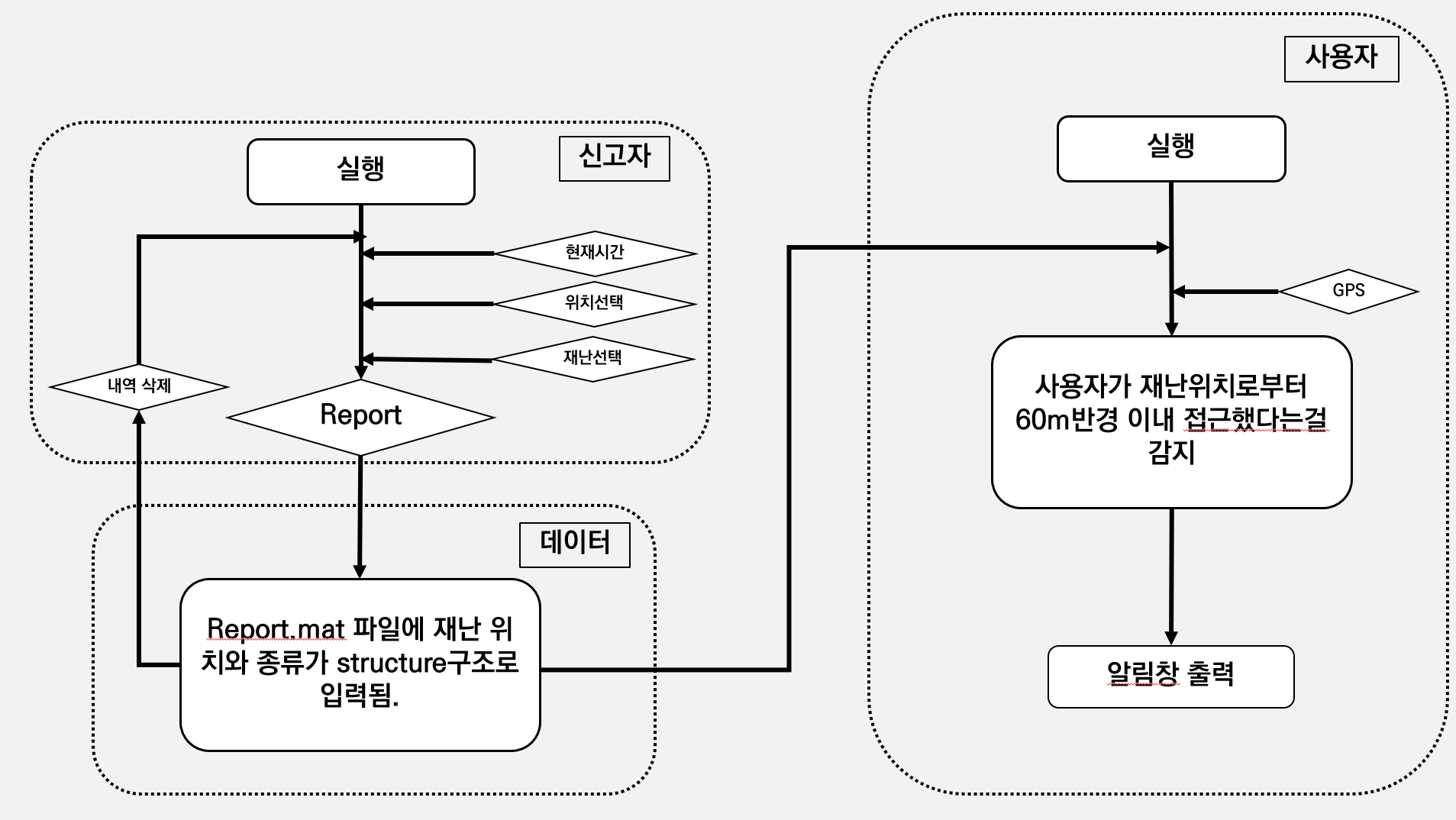
**2.1 배경 지식**

‘재난 알림 시스템‘ 프로그램은 재난 발견자가 프로그램의 신고 툴을 이용해 재난을 신고하면, 다른 사용자들에게 알람을 보내주는 서비스이다. 이러한 시스템은 실제로 이미 존재하지만, 재난 알림의 범위가 너무넓고 그 내용이 상세하고 구체적이지 않아 그 실효성이 적다고 느꼈다. 따라서 우리는 “확실하게 인지되는 재난 알림”을 모토로 프로그램을 설계했다.

신고자가 재난을 신고할 때, 프로그램은 재난의 종류와 발생시각, 신고자의 위치를 서버로 전송한다. 급박한 재난 상황에서 자신이 있는 곳의 정확한 위치를 파악하기란 쉬운 것이 아니기 때문에, 재난의 위치를 자동으로 분석해 전송하도록 했다. 또한 구두신고 시 상황파악에 필요한 필수정보들이 누락 될 경우를 대비해서 각 재난의 상황마다 재난 알림에 필요한 요소들을 GUI를 통해서 선택하도록 하여 필수내용의 누락이 없게 의도했다.

사용자의 위치는 (nmea 로깅) 어플을 이용하여 분석 및 전달받았다. 로깅된 GPS 데이터들 중 GPGGA 정보에 대응하도록 프로그램을 구성했다.

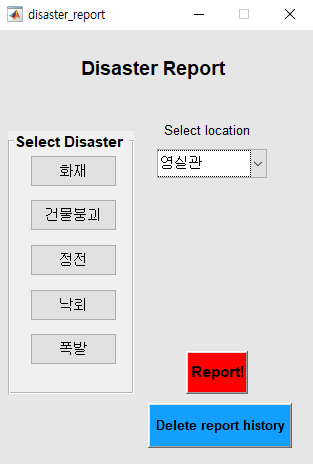
**2.2 프로그램 순서도**



**2.3 소스코드 분석**

**2.3.1 report 코드**

**(1) REPORT GUI의 기본 레이아웃**



재난의 종류,Report와 내역 삭제는 ‘누름 버튼’으로 할당하였고 위치 선택의 경우 건물의 개수가 많은 관계로 ui의 편의성을 위해 ‘팝업 메뉴’로 설정하였다.

**(2) 재난 위치 선택 gui**

function popupmenu1\_Callback(hObject, eventdata, handles)

% hObject handle to popupmenu1 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents as cell array

% contents{get(hObject,'Value')} returns selected item from popupmenu1

str = get(hObject,'String');

val = get(hObject,'Value');

switch str{val}

case '용덕관'

handles.place = [127.07333332231524,37.55131807036886];

handles.name = '용덕관';

case '군자관'

handles.place = [127.07372568918674,37.54958966744157];

handles.name = '군자관';

case '영실관'

handles.place = [127.07333935549572,37.55243147732009];

handles.name = '영실관';

case '충무관'

handles.place = [127.073976,37.552261];

handles.name = '충무관';

case '다산관'

handles.place = [ 127.074189,37.552513];

handles.name = '다산관';

899714321,37.551477390011264];

handles.name = '박물관';

**.....(중략)**

guidata(hObject, handles);

세종대의 대표건물과 그에 해당하는 위도,경도 위치를 ‘팝업메뉴 function’에 switch 구문으로 입력을 하였다.

gui 팝업 메뉴에서 신고를 원하는 위치를 선택후 report버튼을 누르게 되면 handles structure의 place와 name에 기록이 된다.

**(3) 재난 선택**

function pushbutton1\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton1 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

handles.disaster = '화재';

handles.mo = '화장실 오른쪽에 소화기가 비치되어 있습니다.';

guidata(hObject,handles);

% --- Executes on button press in pushbutton2.

function pushbutton2\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton2 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

handles.disaster = '건물붕괴';

handles.mo ='엘리베이터 홀, 계단실 등과 같이 견디는 힘이 강한 벽체가 있는 안전한 곳으로 임시 대피하십시오.';

guidata(hObject,handles);

% --- Executes on button press in pushbutton3.

function pushbutton3\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton3 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

handles.disaster = '정전';

handles.mo ='주변의 비상 손전등을 찾아 시야를 확보하십시오';

guidata(hObject,handles);

**....(중략)**

재난의 종류는 각각의 ‘누름버튼 function’에 할당하였다.

누름 버튼을 누르면 ‘handles structure'의 disaster에 재난의 종류가 기록이 되고, mo에 그 재난에 맞는 행동요령도 같이 기록이 된다.

**(3) 신고 저장역할**

function pushbutton6\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton6 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

t = now; %%(현재 시간-분까지)

d = datetime(t,'ConvertFrom','datenum'); %%(신고시간)

load report.mat %%(신고내역 불러오기) -(1)

di.r.mo = handles.mo; %%(di structor안에 mo가 행동요령)

di.r.time = d; %%(di structor안에 시간 삽입)

di.r.kind = handles.disaster ; %%(di structor안에 재난 종류)

di.r.pl = handles.place ; %%(di structor안에 신고위치 저장)

di.r.na = handles.name; %%(di structor안에 신고위치 건물명)

handles.z = r; -----------)(r to (1) )

handles.z = [handles.z,di];

r = handles.z; (save 목적)

guidata(hObject,handles);

save('report.mat','r')

fprintf('\n신고가 성공적으로 접수되었습니다.\n재난 종류:%s\n발생장소:%s\n신고 날짜 및 시간:%s\n',di.r.kind,di.r.na,d)

report 버튼을 누르면 현재 시간이 데이터 형식으로 d에 저장이 된다.

그 후 report.mat 파일을 불러와 d의 값과 (1),(2)에서 기록한 handles structure를 di.r structure에 기록하여 report.mat 파일에 저장한다.

저장이 정상적으로 완료가 되면

“신고가 정상적으로 접수되었습니다.

재난 종류:

발생장소:

신고날짜 및 시간:“

이 matlab 커맨드창에 출력이 되며 report.mat에 r structure가 저장이 된다.

% --- Executes on button press in pushbutton8.

function pushbutton8\_Callback(hObject, eventdata, handles)

% hObject handle to pushbutton8 (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

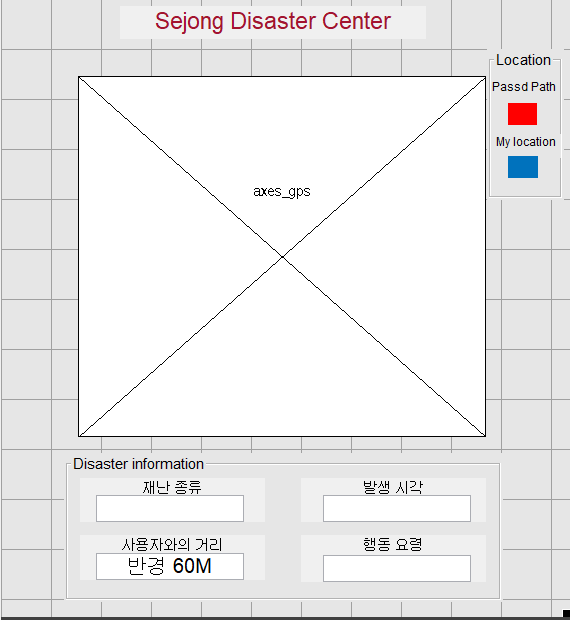
r = [];

save('report.mat','r');

(저장과정 삭제만드는 부분)

**2.3.2 Main window 코드**

**(1) Main window 기본 레이아웃**



axes 창으로 세종대의 지도와 tracking이 그려진다.

이동경로는 빨간색 선으로 표현이 되고 현재 위치는 파란색점으로 표현이 된다.

사용자가 이동을 하는 도중 신고된 재난의 위치에서 반경 60m이내에 접근하게 되면 현재 위치 가 plot되며 텍스트창에서 신고된 재난의 종류,시각,행동요령이 출력된다.

**(2) axes창에 세종대의 지도를 배경으로 삽입하기**

function Mainwindow\_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

% varargin command line arguments to Mainwindow (see VARARGIN)

%%%%% 기본 설정 %%%%% -세종대 지도를 배경화면에 삽입

axes(handles.axes\_gps);

image([127.07220 127.07653], [37.54782 37.55285], imread('map.png'));

% Choose default command line output for Mainwindow

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes Mainwindow wait for user response (see UIRESUME)

% uiwait(handles.figure1);

axes 창을 call하여 (위의 코드의 경우, axes의 tag가 axes\_gps이므로 handles.axes\_gps를 call을 함) x,y축(위도와 경도)의 범위를 설정하고 세종대 지도의 이미지 파일인 ‘map.png'파일을 불러온다.

**(3) nmea 데이터중 GNGGA 데이터만 불러와 위도,경도 데이터 가져오기**

function varargout = Mainwindow\_OutputFcn(hObject, eventdata, handles)

% varargout cell array for returning output args (see VARARGOUT);

% hObject handle to figure

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

format long

%시작했을때 안내창%

Information\_window

%신고한내용 불러오기%

load report.mat

% GPS 데이터 파일 로드 창 %

[fn, pn] = uigetfile('\*.txt', 'GPS 데이터 txt 파일을 불러오세요');

if fn == 0 & pn == 0, errordlg('파일이 선택되지 않았습니다!', '알림'), end

% 데이터 로드 %

cpath = pwd; % 현 주소 저장

cd(pn); % pn 경로 이동

fid = fopen(fn, 'rt'); % fn 파일 로드

cd(cpath); % 주소 복귀

lat = []; long = []; %위도=lat, 경도=long &&(GPS불러오기)

while feof(fid) == 0 &&(GPS불러오기-행렬로 읽는과정)

buffer = fgetl(fid);

if strcmp(buffer(1:6), '$GNGGA') == 1

lat = [lat; buffer(18:28)];

long = [long; buffer(32:43)];

end

end

status = fclose(fid); % .txt 파일 close

% DMM -> DD 좌표 변환 %

lat = str2num(lat(:, 1:2))' + str2num(lat(:, 3:11))'/60;

long = str2num(long(:, 1:3))' + str2num(long(:, 4:12))'/60;

%지도 넘어가는 좌표 데이터 제거 %

i = 1;

while 1

if long(i) < 127.07220 | long(i) > 127.07653 | lat(i) < 37.54782 | lat(i) > 37.55285

lat(i) = []; long(i) = [];

else

i = i + 1;

end

if i == length(lat), break, end

end

NMEA 데이터가 기록된 TXT 파일을 정상적으로 불러오면, TXT의 각각의 줄의 1~6번째 까지의 글자가 ‘$GNGGA’라는 글자와 일치하는지를 확인한다.

만약 조건이 성립한다면, 그 줄의 18~28번째 까지의 문자열을 lat 행렬의 2번째 행에 기록하고, 32~43번째 까지의 문자열을 long 행렬의 2번째 행에 기록한다.

그 후 문자열로 기록된 정보를 수치로 변환하기 위해 str2num함수를 이용한다.

만약 nmea 데이터중 gps가 튀어 비정상적인 위치가 기록되었을 가능성이 있기 때문에, 위의 방법을 통해 얻어진 데이터들 중 특정 범위를 벗어나는 값은 빈행렬로 만들어 tracking에 문제가 없도록 코드를 구성하였다.

**(4) Tracking을 axes창에 구현**

i=1;

while 1

% long과 lat의 값들을 점을 찍어서 plot -즉 long, lat 좌표점.

plot(long(i),lat(i),'r.', 'Markersize', 10); hold on;

drawnow limitrate; %그래프를 업데이트함

(3)에서 얻어낸 lat과 long 행렬의 각n번째 원소들을 좌표로 하여, 빨간색점으로 axes창에 그려진다.

많은 개수의 점이 촘촘히 그려지므로 마치 선을 그린것과 같이 plot된다.

**(5) 사용자와 재난위치 사이의 거리 구하기 & 알림출력**

j=1;

%i번째 x,y좌표와 신고된 재난위치들과의 거리 = R =[첫 번째 신고와의 거리,두 번째 신고,...,마지막신고] (R=[R(1) R(2) R(3)...R(n)])

while j<=length(r)

R(j) = ((long(i)-r(j).r.pl(1,1))^2+(lat(i)-r(j).r.pl(1,2))^2)^0.5; (점과 점사이거리)

if R(j) < 7.863841300448725e-04 % 반경60m.

b = plot(long(i),lat(i),'b.', 'Markersize', 20); (파란색 포인트 잡은것)

(set part = 알림에 따른 신고정보)

set(handles.edit1,'String',sprintf('%s',r(j).r.kind));

set(handles.edit2,'String',sprintf('%s',r(j).r.time));

set(handles.edit4,'String',sprintf('%s',r(j).r.mo));

message = sprintf("%s \n %s에서 '%s'이(가) 발생했습니다. \n\n \*행동요령\*\n %s ",r(j).r.time,r(j).r.na,r(j).r.kind,r(j).r.mo) ;

questdlg(message, '알림', '확인', '확인')

delete(b);

r(j).r.pl=[0,0];%알림이 뜬 이후에 중복 알림이 안뜨기 위해서,신고위치에 0,0을 대입(반경보다 매우큰 거리로 대체)

end

j=j+1;

set(handles.edit1,'String',sprintf(' '));

set(handles.edit2,'String',sprintf(' '));

set(handles.edit4,'String',sprintf(' '));

end

if i == length(lat)

break

end

i=i+1;

end

% Get default command line output from handles structure

varargout{1} = handles.output;

사용자의 위도 경도(lat와 long 행렬)와 신고된 재난위치의 위도 경도(r.pl structure)의 거리를 계속해서 연산하여 R 행렬에 기록한다.

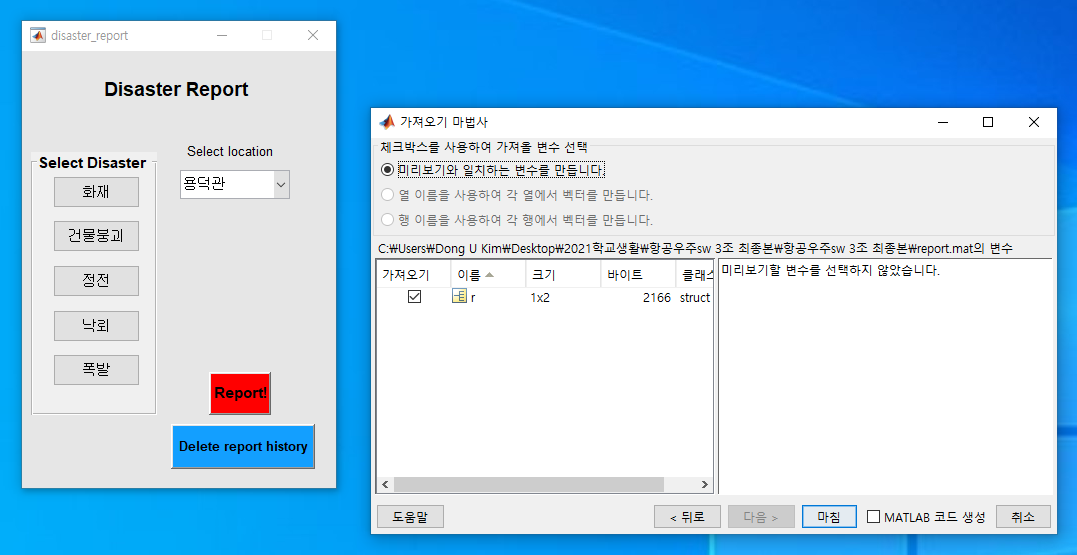
R의 원소중 (60m를 의미) 값보다 작아지면 그 원소의 순서와 같은 lat과 long의 원소를 좌표값으로 불러와 axes창에 파란색점을 그린다.



이때, R(j)의 값이 그대로 있으면 알림창을 꺼도 계속해서 알림이 출력되기 때문에 이를 방지하기 위해 r(j).r.pl structure를 [0,0]행렬로 만들어준다.

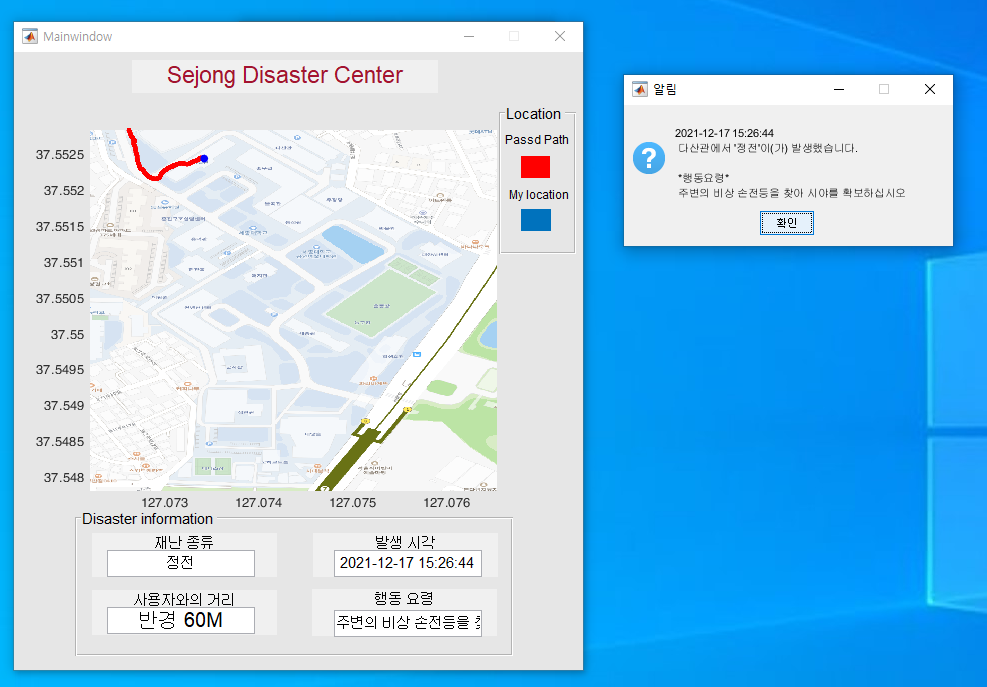
r(j).r.pl structure가 [0,0]행렬이 되면 R(j)의 값이 엄청나게 커지기 때문에 조건에 만족하지 않아 더 이상 출력이 되지않게 된다.

**2.4 결과 및 분석**

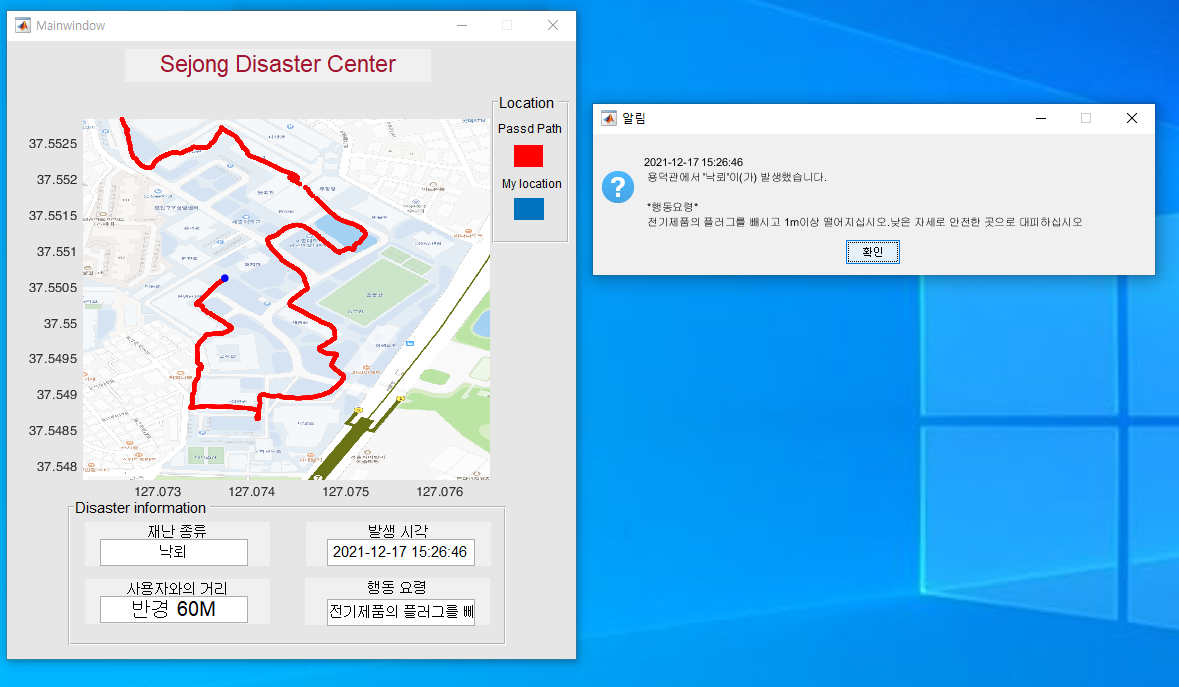


report를 하게 되면 report.mat 파일에 r structure가 정상적으로 기록이 되는 것을 확인 할 수 있었다.

delete report history를 누르면 r structure가 빈행렬이 되어 모든 기록이 사라진다.



nmea txt기록을 정상적으로 불러오면 빨간색 선으로 tracking을 하다 재난 위치 반경 60m이내에 들어왔을 때 현재 위치가 파란색점으로 그려지며 알림창이 출력되는 것을 확인 할 수 있다.



1x2 structure로 기록이 되었기 때문에 위의 그림이후 다른 신고위치에서 알림창이 출력되는 것을 확인 할 수 있다.

코드를 짜며 가장 어려웠던 점이 같은 알림창이 중복으로 출력되는 것을 방지하는 것이 가장 어려운 부분이였다.

이것을 해결하기 위해 알림이 출력된 직후 R행렬의 원소값이 조건을 충족시키지 못하도록 값을 바꾸는 방향으로 결정하였다.

반경이 60m 이내로 접어들게 되는 R 행렬의 원소 위치와 동일한 위치의 r.pl structure를 빈 행렬로 바꿔 R(j)의 값을 바꾸는데 성공하였다.

그 외에 Report 와 Main window가 report.mat파일을 통해 정상적으로 호환이 잘 되었고, nmea txt파일에서 우리 조가 필요로하는 위성데이터만을 추출하여 지도상에 tracking을 구현하는데 성공하였다.

**결론**

**3.1 결론**

지금까지 재난경보 시스템 개념과 작동 방식에 대해 알아보았다. 실행 결과, 데이터의 이동이 txt파일의 형태로 잘 이루어지는 것을 확인하였으며, 또 이 파일에서 GNGGA 위성 데이터만을 추출할 수 있었다.

따라서 신고자와 서버, 수신인 간에 어떠한 데이터를 보내도 서로 확인할 수 있는 토대를 잘 갖추게 되었다.

또한 신고자의 GUI는 누름버튼과 팝업메뉴의 function에 switch구문을 넣음으로써 신고자가 빠르고 간단하게 재난의 정보를 서버로 송신할 수 있었다.

Main window에서는 사용자가 어플리케이션을 실행시키기만 하면 gps데이터를 쉽게 업로드 할 수 있도록 하였고, 정상적으로 txt파일을 불러오면 즉각적으로 tracking이 되어 사용자의 이동경로를 한눈에 쉽게 파악할 수 있게 하였다.

**3.2 향후 발전 방안**

이 프로그램을 통해, 사람들은 기상청 및 재난안전청 등에서 발신하는 국가적 규모의 재난 정보뿐만 아니라, 비교적 작은 재난, 사고 등에 대해서도 정보를 수신하여 피해를 예방할 수 있게 될 것이다. 또한, 공공기관과의 협력을 통해 재난과 사고에 대해 정보를 바로 받을 수 있다면 더 정확한 정보를 더 빠르게 전달할 수 있을 것이다.

**기타**

**4.1 조원 기여도**

- 김동우 : 100%

- 김영섭 : 100%

- 윤태진 : 100%

- 한규동 : 100%

**4.2 참고자료**

**-국민재난안전포털** (<http://www.safekorea.go.kr/idsiSFK/neo/main/main.html>)

**-기상청** (<http://www.weather.go.kr/weather/main.jsp>)

**-Mathworks 함수 가이드** (https://kr.mathworks.com/help/matlab/functionlist.html)

**-지도 출력 관련 자료 --?**