FallingTalk: An IoT-Based Real-Time Falling Detection System

109550151 張昕莓
110550034 孫承瑞
110550143 洪巧芸
110550158 葉家蓁

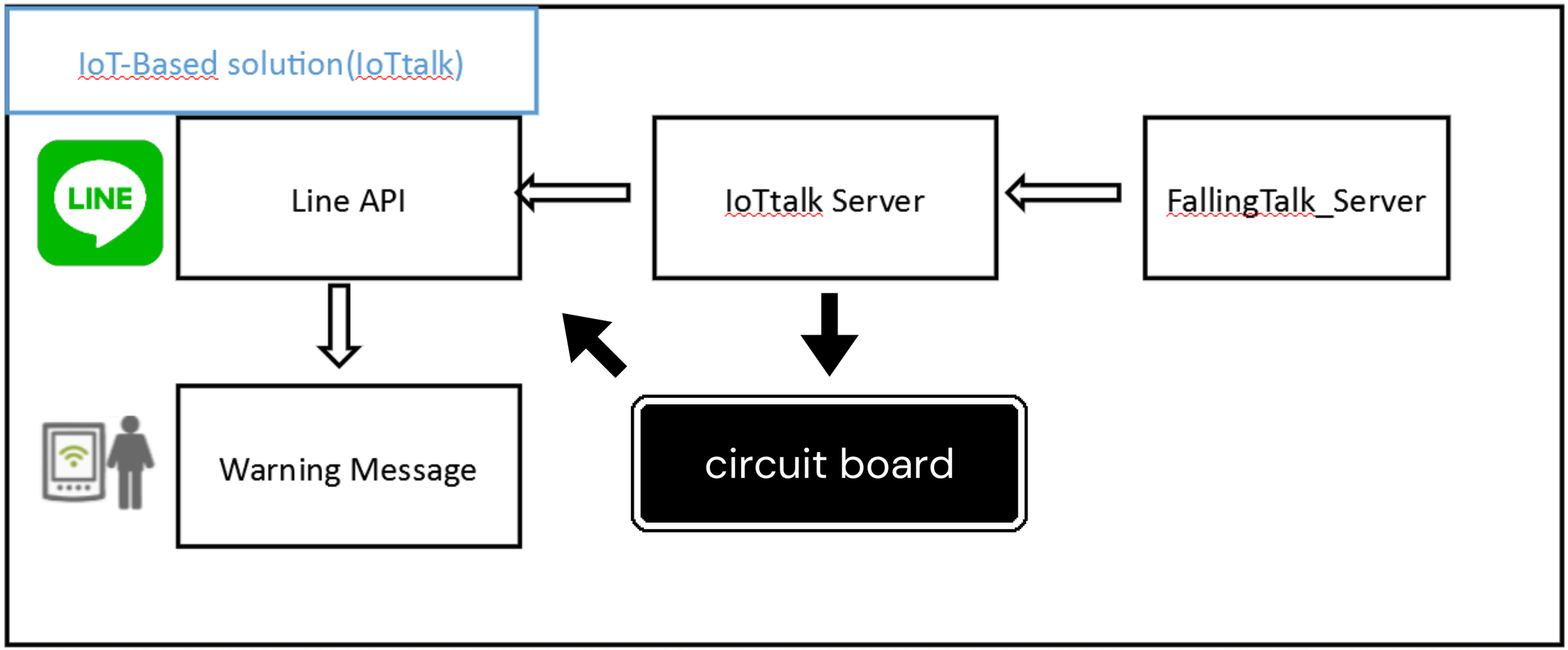
Outline

- ✦ Abstract
- ✦ Motivation
- ✦ Introduction
- ✦ Result

Abstract

- We present a system for detection of human falling from web-cam or video for support of elderly people living alone in their homes.
- In this system, we have the main two parts, which are IoTTalk part and Detection part.
- IoTTalk part, including DAI and Line API.

Overview


IoT-Based solution(IoTtalk)
LINE
Line API
IoTtalk Server
FallingTalk_Server
Warning Message
circuit board

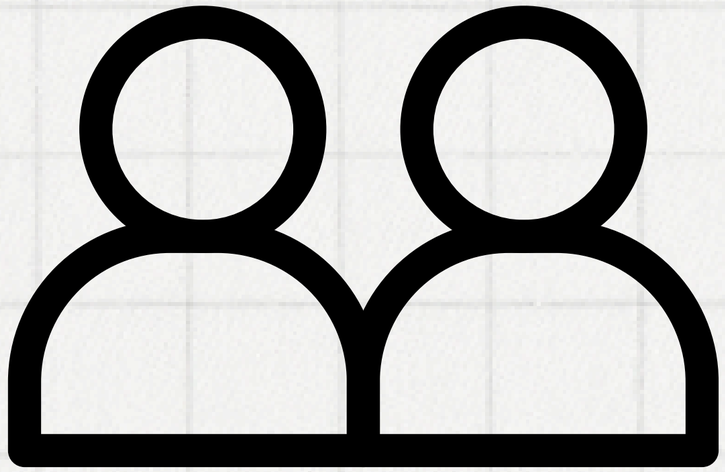
Motivation

With the trend of an aging society in recent years, we need some methods to help elderly people enhance their ability to live independently.

Introduction

How it works ?

Exploring creativity



Human Detection

Falling Detection

Human Detection

HOG +SVM

```
6       # Pre-settings
7       hog = cv2.HOGDescriptor()
8       hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
```

Human Detection : detectMultiScale

```
22          rects, _ = hog.detectMultiScale(frame, winStride=(8,8), padding=(8,8),
23                                                  scale=1.05, useMeanshiftGrouping=False)
```

detect humans in each frame captured from a video feed. It returns rectangles (rects) where it believes humans are present.

Human Detection : result

Falling Detection : backgroundsubtraction

```
42          # Background subtractor
43          fgmask = fgbg.apply(frame)
```

helps isolate moving objects (foreground) from the static background in video frames.

Falling Detection : noise reduction

```
45          # Reduce noise
46          fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_CLOSE, kernel)
47          fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel)
```

performed using morphological operations like closing and opening to clean up the foreground mask, resulting in a clearer view of moving objects.

Falling Detection : contours detection

```
49          # Find contours
50          contours, _ = cv2.findContours(fgmask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

Falling Detection : analylation of contours

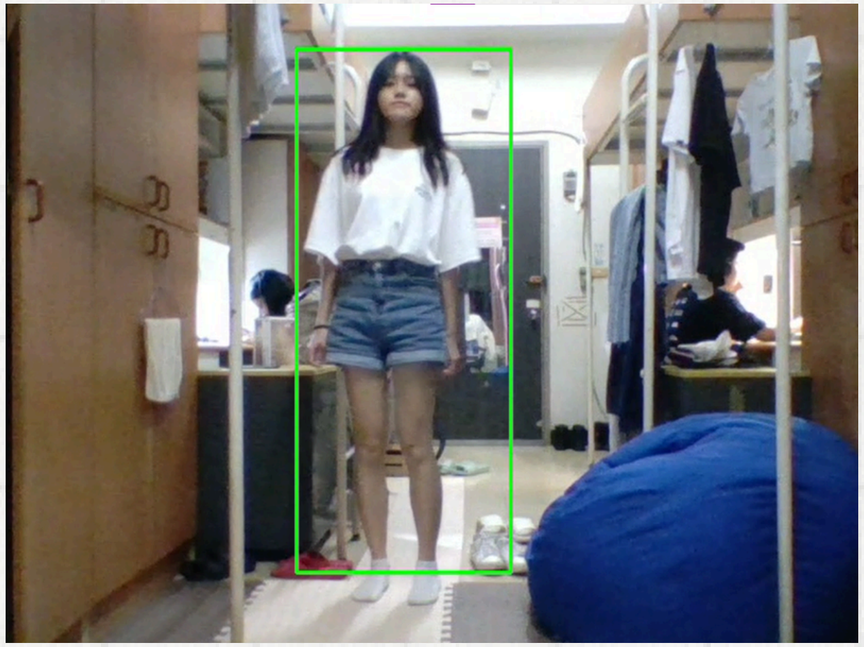
```
52          for contour in contours:
53              if cv2.contourArea(contour) < area_threshold:
54                  continue
55
56              # Turn the contour into a rectangle
57              x, y, w, h = cv2.boundingRect(contour)
58              aspect_ratio = float(w) / h
```

Falling Detection : analylation of contours

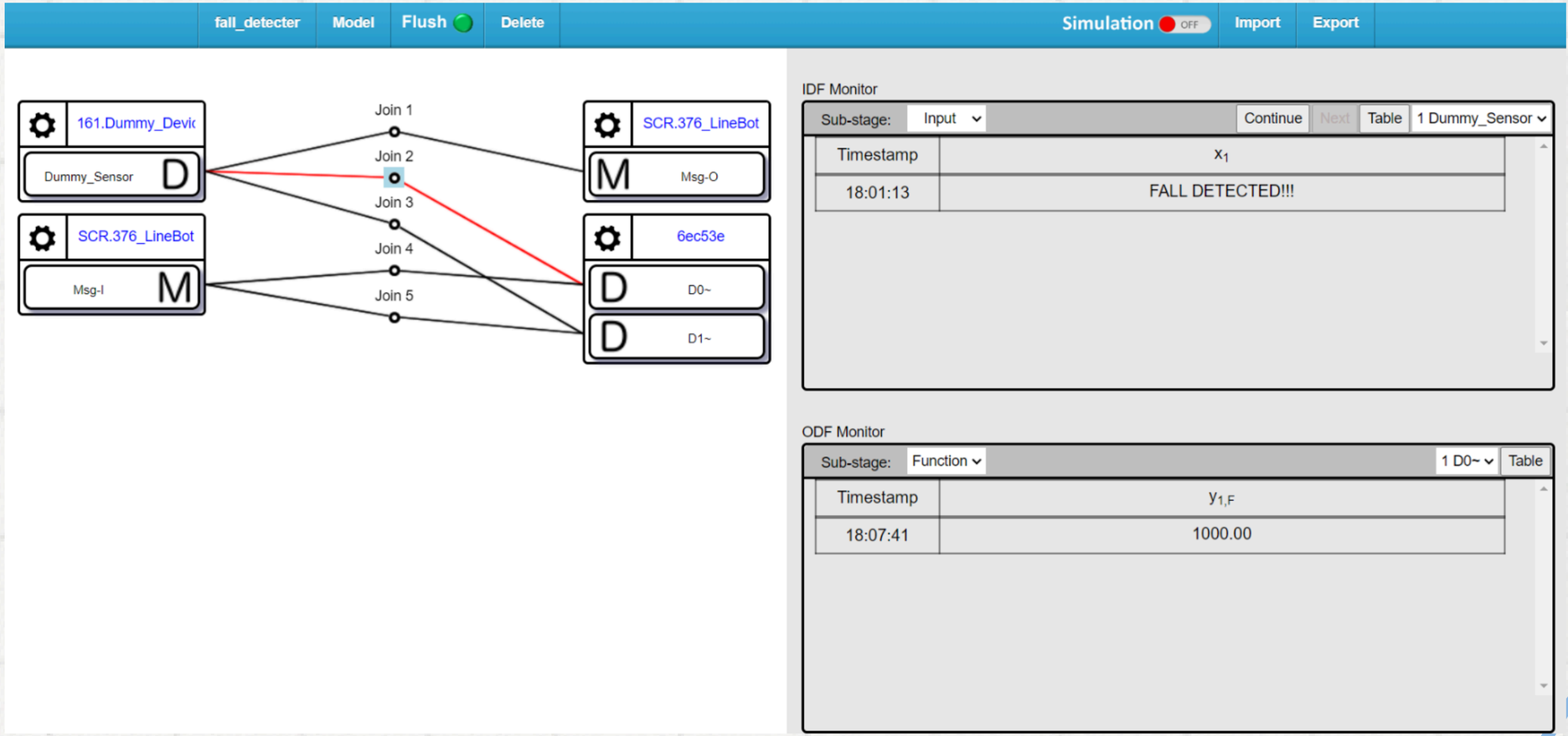
```
60              # Check if fall
61              if aspect_ratio > 1.5:
62                  cv2.putText(frame, 'Fall Detected!', (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)
63                  fall_detected = True
64
65              cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

These contours are analyzed to determine if they match typical fall patterns.

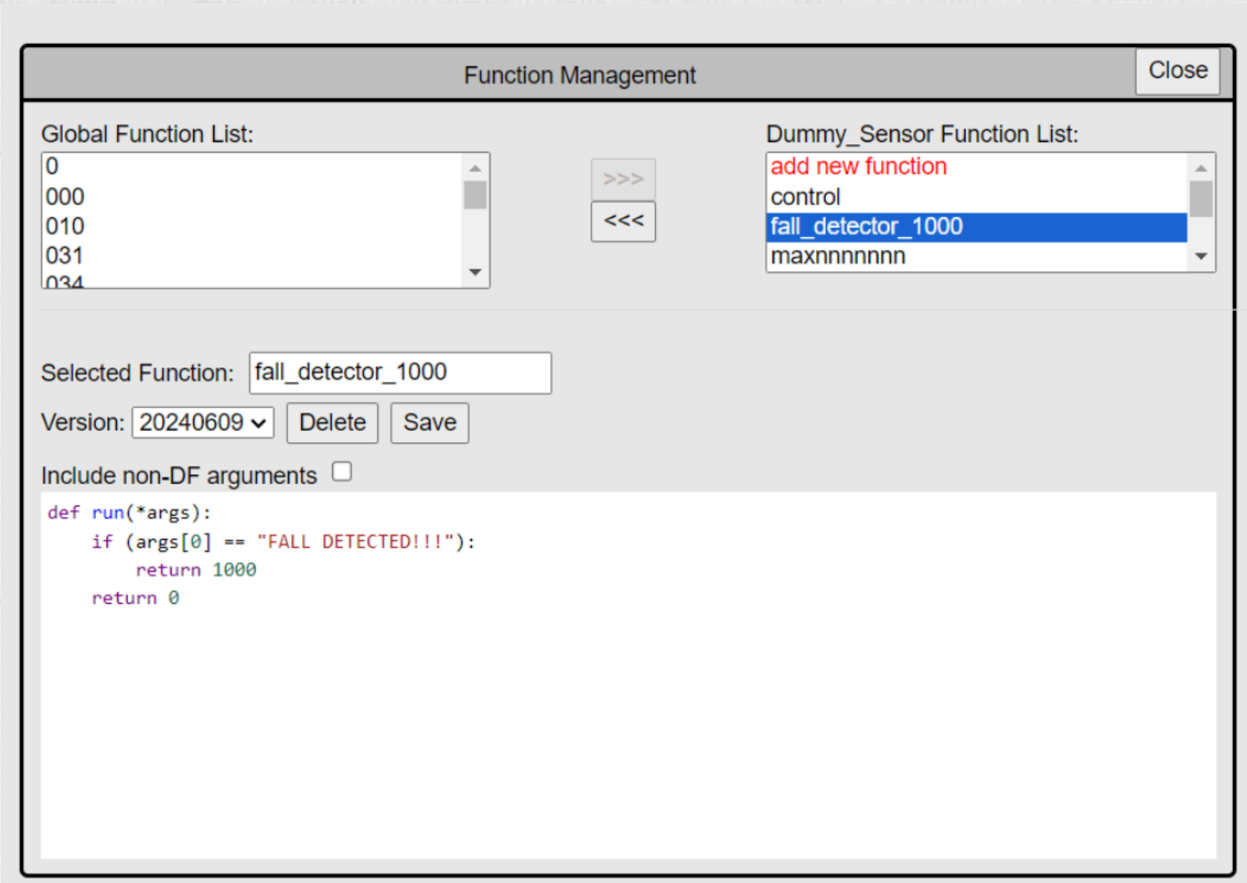
Falling Detection : result



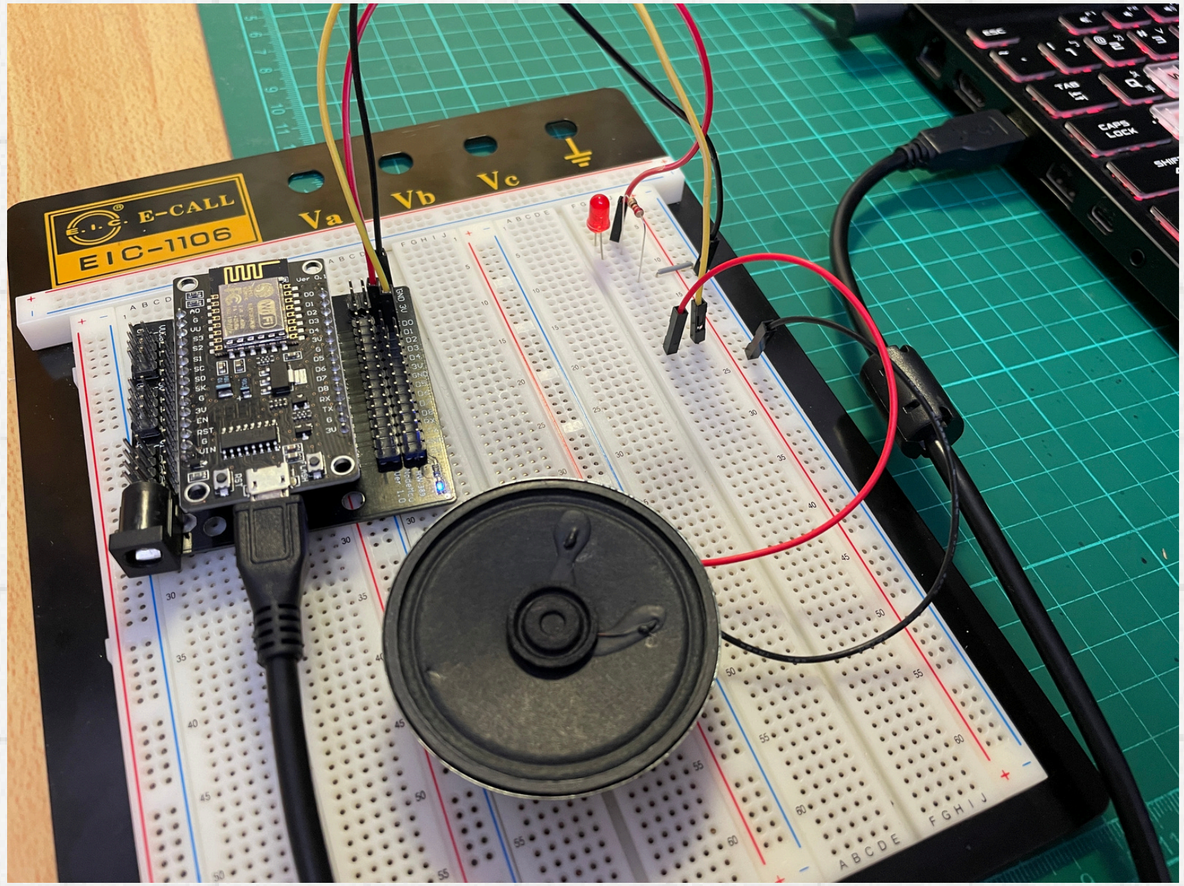
IoTTalk : dummydevice



IoTTalk : dummydevice



circuit board



We use speakers and LED lights to simulate an alarm.

Result

Linebot

```
108         if ("解除" in Msg):
109             DAN.push('Msg-I', [0])
110             line_bot_api.reply_message(event.reply_token,TextSendMessage(text="成功解除警報！"))
111         else:
112             line_bot_api.reply_message(event.reply_token,TextSendMessage(text="若要解除警報請輸入「解除」。"))
```

At the same time, the alarm and LED lights on the circuit board will also stop working.

Demo