**약 처방 및 배달 로봇**

5조 설계 문서

**INDEX**

[**INTRODUCTION**](#_oltzn4o0irtq) **3**

[**SERVICE SCENARIO**](#_i7kzn24hjz9c) **4**

[**REQUIREMENTS**](#_cayw3rpi0gmc) **4**

[**SYSTEM OVERVIEW**](#_9tletmx9mtea) **5**

[**FLOW CHART**](#_7df6eh4mgrj8) **6**

[**PRODUCT FUNCTIONS**](#_jhps0z9avrqv) **7**

[**HARDWARE INTERFACE**](#_ov6ge8qv6qou) **9**

[**APPENDIX [하드웨어 스펙]**](#_vefshtvkiuj4) **9**

[**APPENDIX [병원 내 환자 안전]**](#_2t54pstz2hr5) **11**

[**APPENDIX [그 외 추가 논의 사항]**](#_rbg13yfp51t1) **12**

# **INTRODUCTION**

로봇은 인간의 개입없이 자동적으로 작동하도록 프로그램된 기계로, 핵심 키워드는 ‘자율성’ 이다. 그렇기 때문에 자율주행 자동차, 자율비행 드론 등 물리적인 장치를 갖추고서 컴퓨터 프로그램에 의해 자동적으로 작동함으로써 주어진 작업을 수행하는 기계들은 모두 로봇의 범위에 속한다. 최근, 로봇과 관련된 업계에서 ‘배달 로봇(Delivery Robot)이 화두로 떠오르고 있다. 아마존, 배달의 민족 등 실제 현업에서 배달 로봇을 사용하고 있다. 조사[[1]](#footnote-1)에 따르면 서비스 로봇의 시장 규모는 2018년 112억 달러에서 2023년 297억 달러로 연평균 21.44% 성장할 것으로 예상됐다.

<그림 1. 시도별 전국 의료 인력> <그림 2. 시도별 전국 의료기관 현황>

그림 1과 그림 2로 지역별 의료 인력 편차가 큰 것을 확인할 수 있다. 전체 의료 기관 중 서울에 위치한 의료기관은 12.03%, 인력은 전체 중 서울에 24.89% 위치해 있지만 부산에는 전체 의료 기관 중 10.81%가 위치해 있고, 8.01%로 의료기관의 분포에 비해 의료 종사자의 분포가 적음을 알 수 있다. 이러한 인력 문제를 해결하기 위해 단순 업무들을 대신하는 스마트 의료 서비스 로봇을 제공한다. 그리고 이 서비스를 통해 생기는 여유 인력을 의료 종사자가 부족한 곳에서도 조금 더 원활하게 일이 돌아갈 수 있도록 하고자 한다.

# SERVICE SCENARIO

|  |  |
| --- | --- |
| 구성 요소 | 설명 |
| 사용자 그룹 | 지방에 있는 대학 병원에서 일하는 간호사 |
| 시놉시스 | 간호사 A씨는 지방에 있는 대학병원에 다닌다.  환자 수에 비해 일하는 간호사의 수가 적다.  입원해 있는 많은 환자들에게 시간에 맞추어 약을 전달해야 되는 등의 단순 작업으로 쉴 새 없이 바쁘다.  여유가 없어 환자들에게 질 좋은 서비스를 제공하지 못해 아쉽다. |
| 니즈 | 환자들에게 좀 더 질 좋은 서비스를 제공하고 싶다. |
| 불편사항 | 단순 반복 노동이 많아 시간적 여유가 없다. |
| 대안마련 | 단순 작업을 자동화한 로봇을 이용하여 업무의 효율성을 높인다. |
| 서비스 구성 | 약 배달 로봇 |

# REQUIREMENTS

\*SA=Safety, P=Performance, SE=Security

|  |  |
| --- | --- |
| 요구사항 P-001 | 환자 처방 맞춤형 로봇 주행 |
| 상세 설명 | 환자 맞춤형 약 배달을 위해  1 환자 정보를 서버로부터 내려받는다.  1.1 환자 정보를 내려 받을 때 방문할 다음 병실의 환자 정보만을 받는다.  1.2 방문 완료 시에는 내려 받은 환자 정보를 삭제한다.  2 시간에 맞추어 해당 병실 앞으로 이동  3 해당 환자에게 약을 전달한다 |

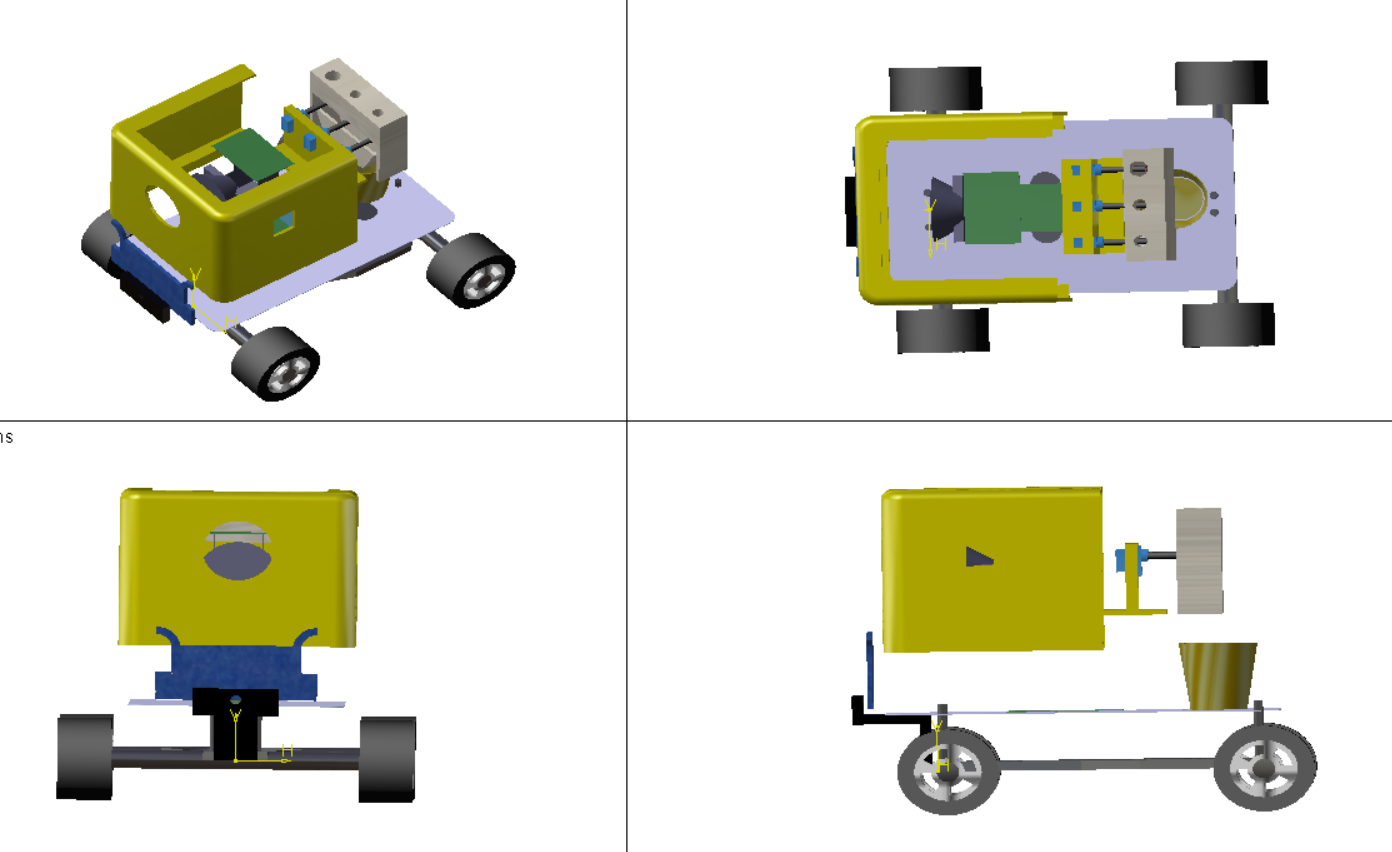
|  |  |
| --- | --- |
| 요구사항 SA-001 | 환자 이중 확인 |
| 상세 설명 | 기존의 환자 확인 절차를 따라  1 개방형 질문을 통한 환자의 참여를 요구  2 최소한 2가지 이상의 지표를 사용하여 환자를 확인 |

|  |  |
| --- | --- |
| 요구사항 SE-001 | 로봇 도난 방지 |
| 상세 설명 | 정해진 시간 외에 일정 시간 정해진 위치에서 벗어났을 경우  1 경고음 발생  2 담당자에게 연락 |

# 

# SYSTEM OVERVIEW

다음 그림 3과 그림 4는 입원 환자에게 약 배달을 하는 모빌리티의 시스템 구성도 이다.



<그림 3. 프로토 타입 로봇>

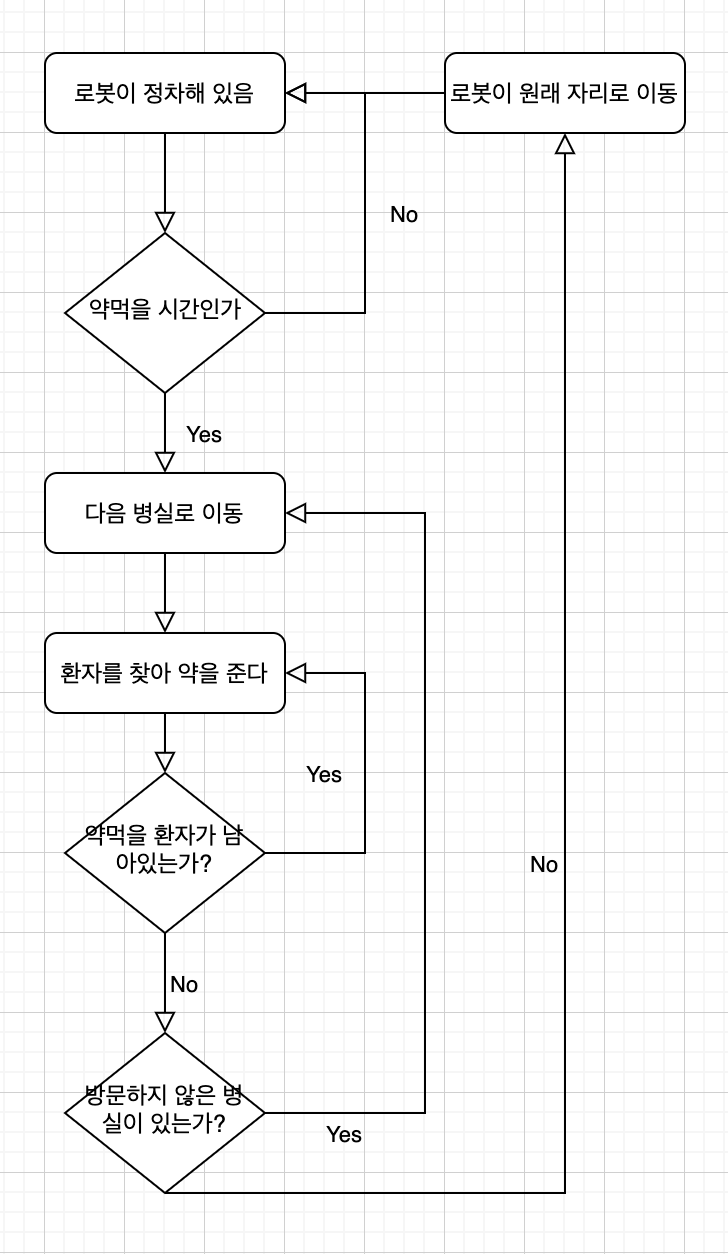
스크린샷이(가) 표시된 사진

자동 생성된 설명

<그림 4. 시스템 개요>

# FLOW CHART

다음 그림은 로봇이 환자에게 약을 배달하기위한 과정을 간략하게 흐름도로 나타낸 것이다. 해당 흐름도에서 나타나 있지 않은 부분은 환자가 병실에 없을 경우, 간호사가 부족한 약을 보충하기 위해 로봇을 호출하였을 경우에 대한 흐름이 빠져있다. 해당 부분은 추후 논의를 통해 추가하고자 한다. 자세한 사항은 APPENDIX [그 외 추가 논의사항]을 참고



<그림 5. 약 배달 흐름도>

# PRODUCT FUNCTIONS

1. 로봇 주행 (서버)

#### server.drive module

|  |
| --- |
| **server.drive.average\_slope\_intercept**(*frame*, *line\_segments*) |
| 영상을 좌우 영역으로 나눠 양쪽 차선의 기울기와 y절편을 구한다. 기울기가 음수일 경우 x1, x2 모두 왼쪽 영역에 있을때 왼쪽 차선, 기울기가 음수가 아닐 경우 x1, x2 모두 오른쪽 영역에 있을때 오른쪽 차선으로 간주한다. 이때 x1, x2가 같을 경우 수직이기에 건너뛴다.   |  |  | | --- | --- | | Parameters: | * **frame** – 웹캠으로 입력받은 영상 * **line\_segments** – 검출한 차선 윤곽선 | | Returns lane\_lines: | 양측 차선 | |

|  |
| --- |
| **server.drive.detect\_edges**(*frame*) |
| 영상의 edge를 찾는 함수이다. 영상을 흑백 이진화 처리한 후 bilateral filter를 이용하여 노이즈를 제거한다. Canny Algorithm을 이용해 edge를 찾는다.   |  |  | | --- | --- | | Parameters: | **frame** (*numpy.ndarray*) – 웹캠으로 입력받은 edge를 찾을 영상 | | Returns numpy.ndarray edges: | 영상의 edge | |

|  |
| --- |
| **server.drive.detect\_line\_segments**(*cropped\_edges*) |
| 관심영역으로 자른 영상을 받아 차선으로 예상되는 선을 검출한다. 입력 영상에서 임의의 점을 대상으로 차선을 검출하는 확률적 허프변환을 이용하여 차선을 찾는다.   |  |  | | --- | --- | | Parameters: | **cropped\_edges** – 차선을 검출할 이진화된 영상 | | Returns line\_segments: | 검출한 차선 윤곽선 | |

|  |
| --- |
| **server.drive.display\_heading\_line**(*frame*, *steering\_angle*, *line\_color=(0*, *0*, *255)*, *line\_width=5*) |
| 차량의 steering angle을 직선으로 표시한다.   |  |  | | --- | --- | | Parameters: | * **frame** – 웹캠으로 입력받은 영상 * **steering\_angle** – 차량의 steering angle * **line\_color** – 표시할 직선의 색 * **line\_width** – 표시할 직선의 두께 | | Returns heading\_image: | steering angle이 표시된 영상 | |

|  |
| --- |
| **server.drive.display\_lines**(*frame*, *lines*, *line\_color=(0*, *255*, *0)*, *line\_width=6*) |
| 영상에 차선을 표시한다.   |  |  | | --- | --- | | Parameters: | * **frame** – 웹캠으로 입력받은 영상 * **lines** – 차선 * **line\_color** – 차선의 색 * **line\_width** – 차선 두께 | | Returns line\_image: | 차선이 표시된 영상 | |

|  |
| --- |
| **server.drive.get\_steering\_angle**(*frame*, *lane\_lines*) |
| 웹캠에서 영상을 입력받아 steering angle을 구한다. 차량이 양측 차선의 중앙으로 가도록 제어한다. 입력 받은 영상에서 검출한 차선의 수에 따라 구한다.   * x\_offset : 양측 차선의 중간(가로축)과 화면 중앙의 차이 * y\_offset : 높이의 절반 * 차선이 2개인 경우 : 두 차선의 중앙으로 향하도록 한다. * 차선이 1개인 경우 : 검출한 차선과 평행하도록 한다.  |  |  | | --- | --- | | Parameters: | * **frame** – 웹캠으로 입력받은 영상 * **lane\_lines** (*int*) – 차선의 수 | |

|  |
| --- |
| **server.drive.make\_points**(*frame*, *line*) |
| 웹캠 영상을 입력받아 좌표를 생성한다. 기울기와 y절편으로 x좌표를 찾는다. 이때, 기울기가 0인경우 0으로 나눌 수 없기 때문에 작은 값인 0.1로 대신 계산한다.   * x1 : 차선의 왼쪽 가로 좌표 * x2 : 차선의 오른쪽 가로 좌표 * y1 : 영상 바닥 세로 좌표 * y2 : 영상 높이의 중간 세로 좌표  |  |  | | --- | --- | | Parameters: | * **frame** – 웹캠으로 입력받은 영상 * **line** – 왼쪽 또는 오른쪽 차선 | | Returns: |  | |

|  |
| --- |
| **server.drive.region\_of\_interest**(*edges*) |
| 입력받은 영상의 하단 절반을 관심 영역으로 자른다. 영상과 같은 크기의 array를 생성한 후 영상의 관심 영역 부분을 복사하여 반환한다.   |  |  | | --- | --- | | Parameters: | **edges** – 관심 영역을 지정할 edge를 찾은 영상 | | Returns cropped\_edges: | 복사한 관심 영역 영상 | |

1. 환자 식별 - 안면 인식 (클라이언트)

client.vision.cascade.haar module

|  |
| --- |
| **client.vision.cascade.haar.find\_face**() |
| 실시간으로 얼굴을 찾아서 반환함   |  |  | | --- | --- | | Returns numpy.ndarray: | 찾아진 그레이 스케일 된 얼굴 이미지 데이터 | |

|  |
| --- |
| **client.vision.cascade.haar.get\_gray\_face**(*frame*) |
| 그레이 스케일된 이미지에서 얼굴부분을 크롭해 반환한다.   |  |  | | --- | --- | | Parameters: | **frame** (*numpy.ndarray*) – 얼굴을 찾고자 하는 그레이스케일된 이미지 파일 | | Returns numpy.ndarray: | 찾아진 얼굴 | |

###### client.vision.recognizer module

|  |
| --- |
| **client.vision.recognizer.find\_patient**() |
| 환자의 얼굴을 찾고 아이디를 반환한다. ymls 폴어 안에 있는 가장 최근의 학습 파일을 이용하여 환자의 얼굴을 찾는다.   |  |  | | --- | --- | | Returns str: | 찾은 얼굴의 아이디 반환 환자가 아닐경우 빈 스트링값을 반환한다. | |

###### client.vision.register module

|  |
| --- |
| **client.vision.register.train**(*patient\_id*, *data\_path*) |
| 환자의 얼굴을 인식시키기 위해 아이디별로 이미지를 학습시킨다.   |  |  | | --- | --- | | Parameters: | * **patient\_id** (*str*) – 학습시킬 환자의 아이디 * **data\_path** (*str*) – 학습시킬 환자의 이미지가 저장되어있는 경로 | | Returns bool: | 주어진 데이터에서 찾아진 얼굴 데이터와 라벨값을 이용하여 train\_recognizer함수를 통해 학습시킨 결과 값 | |

|  |
| --- |
| **client.vision.register.train\_recognizer**(*datasets*, *recognizer\_path='/Users/nuguni/Downloads/project/src/client/vision/ymls/2020-02-21.yml'*, *old\_recognizer=False*) |
| 데이터를 바탕으로 얼굴 정보를 학습시킨다. 학습된 YML파일은 recognizer\_path에 저장되며 OLD\_RECOGNIZER에 따라 이전 파일에 추가할 수 있다.   |  |  | | --- | --- | | Parameters: | * **datasets** (*dict*) – 학습시킬 데이터셋 * **recognizer\_path** (*str*) – 학습된 YML파일을 저장할 경로 * **old\_recognizer** (*bool*) – 이전파일에 추가적으로 학습시킬때 사용되는 flag이값이 참일 경우 recognizer\_path는 자동으로 가장 최근의 YML파일의 경로로 바뀐다. | | Returns bool: | 얼굴 데이터와 라벨을 이용하여 이미지를 학습시킨 결과 | |

###### client.vision.utils module

|  |
| --- |
| **client.vision.utils.get\_latest\_yml\_path**() |
| 가장 최근의 YML 파일의 경로를 가져온다.   |  |  | | --- | --- | | Returns str: | 가장 최근의 YML 파일의 경로 | |

1. 환자 식별 - 음성 인식 (클라이언트)

###### client.voice.stt module

|  |
| --- |
| **client.voice.stt.clova\_stt**(*client\_id='example'*, *client\_secret='example'*, *lang='Kor'*) |
| 네이버 클로바 클라우드 서비스를 이용한 STT 함수 voices폴더 아래에 랜덤 파일이름으로 음성(wav)이 저장된다.  API : <https://apidocs.ncloud.com/ko/ai-naver/clova_speech_recognition/stt/>  price : 4won/15sec   |  |  | | --- | --- | | Parameters: | * **client\_id** (*str*) – 클로바 STT 사용에 필요한 client\_id (default값은 config에 저장 가능) * **client\_secret** (*str*) – 클로바 STT 사용에 필요한 client\_secret (default값은 config에 저장 가능) * **lang** (*str*) – 사용할 언어 ( Kor, Jpn, Eng, Chn ) (default값은 config에 저장 가능) | | Returns: |  | |

|  |
| --- |
| **client.voice.stt.is\_korean\_chr**(*chr\_u*) |
| check if korean characters see [http://www.unicode.org/reports/tr44/#GC\_Values\_Table](http://www.unicode.org/reports/tr44/" \l "GC_Values_Table) |

###### client.voice.tts module

|  |
| --- |
| **client.voice.tts.clova\_tts**(*text*, *client\_id='example'*, *client\_secret='example'*, *lang='Kor'*) |
| 네이버 클라우드 서비스를 이용한 TTS 함수 voices폴더 아래에 랜덤 파일이름으로 다운받은 음성(mp3)을 저장하고 음성 파일을 실행한다. API : <https://apidocs.ncloud.com/ko/ai-naver/clova_speech_synthesis/tts/> price : 4won/15sec   |  |  | | --- | --- | | Parameters: | * **text** (*str*) – 말하고자 하는 문자 * **client\_id** (*str*) – 클로바 TTS 사용에 필요한 client\_id (default값은 config에 저장 가능) * **client\_secret** (*str*) – 클로바 TTS 사용에 필요한 client\_secret (default값은 config에 저장 가능) * **lang** (*str*) – 사용할 언어 ( Kor, Jpn, Eng, Chn ) (default값은 config에 저장 가능) | | Returns: | bool타입의 함수 실행 결과 | |

|  |
| --- |
| **client.voice.tts.say**(*text*) |
| AIY에서 제공하는 기본 TTS   |  |  | | --- | --- | | Parameters: | **text** (*str*) – 말하고자 하는 문자 | | Returns: | bool타입의 함수 실행 결과 | |

###### client.voice.utils module

|  |
| --- |
| **client.voice.utils.play\_mp3**(*file\_path*) |
| mp3형식의 오디오 파일을 mpg123를 이용해 실행한다.   |  |  | | --- | --- | | Parameters: | **file\_path** (*str*) – 오디오 파일의 경로 | | Returns: | bool타입의 함수 실행 결과 | |

|  |
| --- |
| **client.voice.utils.randome\_file\_name**(*length*) |
| 랜덤 문자열인 파일 이름을 생성한다.   |  |  | | --- | --- | | Parameters: | **length** (*int*) – 파일 이름 길이 | | Returns: | 랜덤 문자열 파일 이름 | |

|  |
| --- |
| **client.voice.utils.voice\_recoder**(*file\_name*) |
| voices 폴더에 file\_name으로 음성 파일을 녹음해 저장한다. 버튼을 누르면 음성 녹음이 종료된다.   |  |  | | --- | --- | | Parameters: | **file\_name** (*str*) – 음성 파일을 저장할 파일 이름 | | Returns: | 음성 파일의 절대 경로, 실행 실패시 빈 문자열을 리턴한다. | |

1. 환자 정보 및 처방 약 데이터 베이스 (클라이언트)

###### client.db.database module

|  |
| --- |
| **client.db.database.create\_new\_id**() |
| config파일에 지정되어있는 ID길이에 맞는 환자 ID 생성   |  |  | | --- | --- | | Returns str: | 문자형 환자 아이디 | |

|  |
| --- |
| **client.db.database.create\_random\_number**(*length*) |
| 랜덤한 숫자(문자열)를 생성해내는 함수 입니다.   |  |  | | --- | --- | | Parameters: | **length** (*int*) – 랜덤 숫자의 길이를 결정 짓는 int | | Returns str: | 문자열 타입의 랜덤 숫자 | |

|  |
| --- |
| **client.db.database.create\_random\_string**(*length*) |
| 랜덤한 문자열을 생성해내는 함수 입니다.   |  |  | | --- | --- | | Parameters: | **length** (*int*) – 랜덤 숫자의 길이를 결정 짓는 int | | Returns str: | 랜덤 문자열 | |

|  |
| --- |
| **client.db.database.delete\_medicine\_info**(*patient\_id*) |
| 데이터베이스에서 환자 약 정보 삭제   |  |  | | --- | --- | | Parameters: | **patient\_id** (*str*) – 삭제하고자 하는 환자의 아이디 | | Returns bool: | bool타입의 함수 실행 결과 | |

|  |
| --- |
| **client.db.database.delete\_patient\_info**(*patient\_id*) |
| 데이터베이스에서 환자 정보 삭제   |  |  | | --- | --- | | Parameters: | **patient\_id** (*str*) – 삭제하고자 하는 환자의 아이디 | | Returns bool: | bool타입의 함수 실행 결과 | |

|  |
| --- |
| **client.db.database.get\_medicine\_info**(*patient\_id*) |
| 데이터베이스에 저장되어있는 환자의 약 정보를 가져오는 함수   |  |  | | --- | --- | | Parameters: | **patient\_id** (*str*) – 검색하고자 하는 환자의 아이디 | | Returns dict: | 성공시 약 정보를 담은 dict값, 실패시 필드 네임만 들어있는 빈 dict | |

|  |
| --- |
| **client.db.database.get\_patient\_info**(*patient\_id*) |
| 데이터베이스에 저장되어있는 환자의 정보를 가져오는 함수   |  |  | | --- | --- | | Parameters: | **patient\_id** (*str*) – 검색하고자 하는 환자의 아이디 | | Returns dict: | 성공시 환자 정보를 담은 dict값, 실패시 필드 네임만 들어있는 빈 dict | |

|  |
| --- |
| **client.db.database.has\_patient\_id**(*patient\_id*) |
| 데이터베이스에 환자의 정보가 있는 지 확인하는 함수   |  |  | | --- | --- | | Parameters: | **patient\_id** (*str*) – 검색하고자 하는 환자의 아이디 | | Returns bool: | bool타입의 검색 결과 | |

|  |
| --- |
| **client.db.database.save\_medicine\_info**(*patient\_id*, *medicine\_info*) |
| 새로운 환자의 약 정보를 데이터베이스에 저장 필드값은 ID, medicine1, medicine2, medicine3이다.   |  |  | | --- | --- | | Parameters: | * **patient\_id** (*str*) – 저장하고자 하는 환자의 아이디 * **medicine\_info** (*dict*) – 환자의 약 정보 | | Returns bool: | bool 타입의 실행 결과 | |

|  |
| --- |
| **client.db.database.save\_patient\_info**(*patient\_id*, *patient\_info*) |
| 새로운 환자의 정보를 데이터베이스에 저장   |  |  | | --- | --- | | Parameters: | * **patient\_id** (*str*) – 저장하고자 하는 환자의 아이디 * **patient\_info** (*dict*) – 환자 정보 | | Returns bool: | bool 타입의 실행 결과 | |

1. 서보 제어 (클라이언트)

###### client.sensor.request module

|  |
| --- |
| **client.sensor.request.get\_gpio\_pin\_function**() |
| 모터를 제어하는 서버측 GPIO 핀의 설정 값을 갖고옴   |  |  | | --- | --- | | Returns str: | 서버에 요청 결과 (out, in, 빈스트링 3가지) | |

|  |
| --- |
| **client.sensor.request.gpio\_pin\_change\_in**() |
| 모터를 제어하는 서버측 GPIO 핀을 IN으로 바꿈 설정은 config파일에서 할 수 있음 [SERVER\_IP\_ADDR, SERVER\_PORT, MOTOR\_STOP\_PIN\_NUM]   |  |  | | --- | --- | | Returns bool: | 서버에 요청 결과 | |

|  |
| --- |
| **client.sensor.request.gpio\_pin\_change\_out**() |
| 모터를 제어하는 서버측 GPIO 핀을 OUT으로 바꿈 설정은 config파일에서 할 수 있음 [SERVER\_IP\_ADDR, SERVER\_PORT, MOTOR\_STOP\_PIN\_NUM]   |  |  | | --- | --- | | Returns bool: | 서버에 요청 결과 | |

|  |
| --- |
| **client.sensor.request.is\_webserver\_up**() |
| jsonrpc 서버와의 연결을 확인하는 함수   |  |  | | --- | --- | | Returns bool: | 서버와 연결이 되어있으면 True 아님 False를 반환한다. | |

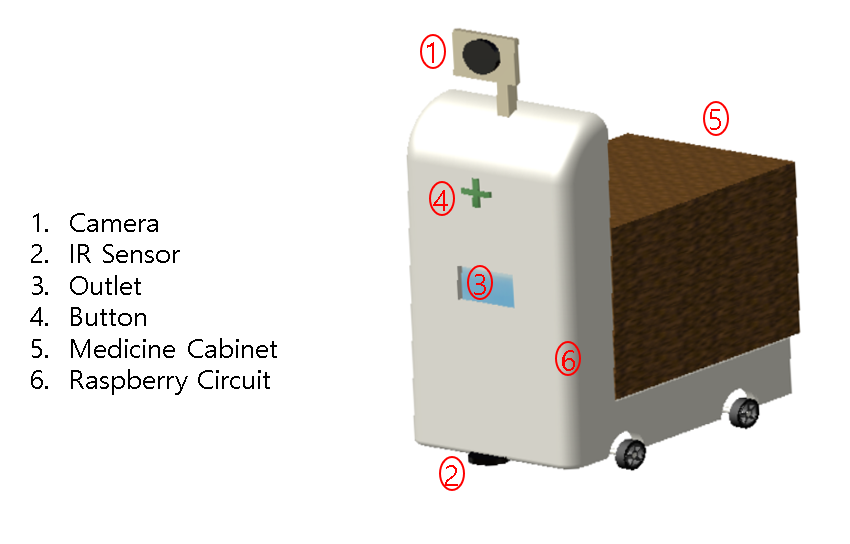
###### client.sensor.servomotor module

|  |
| --- |
| **client.sensor.servomotor.control\_servo\_motor**(*medicine\_name*, *times*) |
| 약의 이름과 횟수에 맞게 서보모터를 작동시켜 약을 배출한다.   |  |  | | --- | --- | | Parameters: | * **medicine\_name** (*str*) – 약의 이름 * **times** (*int*) – 약 배출 횟수 | | Returns bool: | bool형의 함수 실행 결과 | |

|  |
| --- |
| **client.sensor.servomotor.medicine\_out**(*medicine\_info*) |
| 약 정보(medicine\_info)에 맞는 약을 배출   |  |  | | --- | --- | | Parameters: | **medicine\_info** (*dict*) – 약이름과 섭취해야하는 약의 개수가 담겨있다. | | Returns bool: | bool형의 함수 실행결과 | |

# HARDWARE INTERFACE

이 로봇은 프로토 타입으로 안면인식과 병원 내 주행을 위한 카메라, 어두운 환경에서 카메라 센서의 에러를 줄이기 위한 IR 센서, 약을 보관하는 약품 보관함, 약통을 열기 위한 버튼, 이러한 센서들을 제어하기 위한 라즈베리 파이가 탑재되어 있다. 다음 그림 6을 통해 해당 센서들과 부품의 위치를 확인할 수 있다.



<그림 6. 하드웨어 설계도>

# APPENDIX [하드웨어 스펙]

* Camera (Wide Angle Raspberry Pi Camera)

카메라로 영상을 받아와 Open CV로 영상처리 및 추가적인 머신러닝을 이용하여 안면인식을 한다.



< SainSmart Wide Angle Fish-Eye Camera Lenses for Raspberry Pi Arduino >

* Outlet

환자의 처방약을 서버에서 받아와 제어부에서 제공한 약을 환자가 받을 수 있는 출입구이다. 버튼을 누르면 약 캐비넷을 볼 수 있다. 의료진은 환자가 필요한 약을 분류하여 배치시킨다.



* Raspberry Pi Circuit

메인보드로서 총 제어를 담당한다.



* SoC: Broadcom BCM2837B0 quad-core A53 (ARMv8) 64-bit @ 1.4GHz
* GPU: Broadcom Videocore-IV
* RAM: 1GB LPDDR2 SDRAM
* Networking: Gigabit Ethernet (via USB channel), 2.4GHz and 5GHz 802.11b/g/n/ac Wi-Fi
* Bluetooth: Bluetooth 4.2, Bluetooth Low Energy (BLE)
* Storage: Micro-SD
* GPIO: 40-pin GPIO header, populated
* Ports: HDMI, 3.5mm analogue audio-video jack, 4x USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)
* Dimensions: 82mm x 56mm x 19.5mm, 50g

# 

# APPENDIX [병원 내 환자 안전]

1. 환자 안전

가. 개념

환자가 사고 손상으로부터 자유로운 상태 및 의료과오와 위해사건의 발생을 최소화하여 환자가 위해사건에 빠지지 않게 예방

나. 환자안전의 중요성

1) 의료기관은 환자에게 안전하고 질 높은 서비스를 제공할 책임과 의무가 있으며, 병원 내에서 이루어지는 모든 과정에서 환자안전이 제일 우선으로 고려되어야 함

2) 병원 내 환자안전사고 발생시

가) 환자의 생명과 직결되는 치명적 결과 발생

나) 의료의 질 저하

다) 병원의 재정적 손실 발생

라) 원상회복이 어려움

마) 파급효과가 큼

2. 정확한 환자확인

가. 목적 모든 진단과 치료과정에서 환자를 정확히 확인하여 환자에게 안전하고 정확한 치료와 서비스 제공

나. 방법

1) 개방형 질문 - “성함이 어떻게 되십니까?”

2) 최소한 두 가지 이상의 지표(indicator)사용

가) 신분증이나 진료예약증, 진료카드 등에 기재된 환자이름과 등록번호 확인

나) 등록번호 확인이 어려운 경우 생년월일로 확인

3) 환자의 병실호수나 위치를 알리는 지표는 환자확인 지표로 사용불가

4) 모든 상황과 장소에서 일관된 환자확인 방법 적용

5) 환자가 의식이 없거나 의사소통이 불가능한 경우

– 보호자에게 개방형으로 질문하여 확인

다. 시점

1) 의약품 투여 전 2) 혈액제제 투여 전 3) 검사 시행 전 4) 진료 전 5) 처치 및 시술 전

# 

# APPENDIX [추후 보완 부분]

* 장애물 인식 불가

NASA가 주행 중 앞에 있는 장애물을 탐지 하지 못한다는 문제가 있습니다. 이 문제는 앞으로 다음과 같은 방법을 통해 해결될 수 있습니다.

1. 초음파 센서 혹은 적외선 센서를 이용한 장애물 탐지
2. LiDAR를 사용한 장애물 탐지

* 약 전달을 불가 / 투여 거부

NASA가 약을 전달하러 왔을 때 외부에 있거나 기타 다른 이유로 약을 받을 수 없는 상황에 대한 처리가 필요합니다. 이 문제는 다음과 같은 방법으로 해결할 수 있습니다.

1. 약 보관 상자를 이용한 약 전달 방법

환자 개개인 마다 약을 보관할 수 있는 상자를 두어 전달 할 수 있습니다. 이 방법의 경우 환자가 약을 전달을 받을 수 있으나 이를 제대로 확인하기 어려운 환자의 경우 약을 제 시간에 섭취하지 못할 수 있다는 문제가 있습니다.

1. 환자 정보 데이터베이스 서버에 표시

NASA가 약을 제대로 전달하지 못했을 경우 간호사가 다시 전달할 수 있도록 혹은 이를 인지할 수 있도록 서버에 정보를 업데이트하여 간호사에게 알릴 수 있습니다.

* 환자 인식 방법의 다양성

환자를 인식하는 방법에 있어 얼굴인식 방법을 채택하여 기획하였습니다. 하지만 병실에 누워있어야만 하는 환자의 경우 쉽게 얼굴인식을 하기 어렵습니다. 이와 같은 문제를 해결하기 위해 얼굴인식 이외의 방법들이 동원되어야 합니다.

1. 환자가 착용하고 있는 인식표를 이용한 환자 인식

병원내 환자 안전을 위한 문서를 확인해보았을 경우 정확한 환자 확인을 위해 최소한 두 가지 이상의 지표를 사용할 것이라 알리고 있습니다. 그 방법으로 신분증이나 진료예약증, 진료카드 등에 기재된 환자이름과 등록번호 확인 할 수 있습니다. 환자가 갖고 있는 인식표를 통해 환자를 인식할 수 있도록 추가 할 예정입니다.

1. https://www.researchandmarkets.com/research/qfbt9n/top\_robotics?w=12 [↑](#footnote-ref-1)