3주차 요약본

작성자 이의진(엘텍공과대학 휴먼기계바이오공학부)

Chapter 06 비전 에이전트

비전 에이전트: 비전 프로그램 (+ 사용자 인터페이스) ⇒ **환경과 상호작용**

6.1 지능 에이전트로서 비전 에이전트

지능 에이전트

- 최적의 의사결정을 하려 노력한다.
- 센서를 통해 환경을 지각하고 액츄에이터를 통해 환경에 행동을 가한다고 볼 수 있는 모든 것.
- 비전에 특화된 지능 에이전트가 비전 에이전트

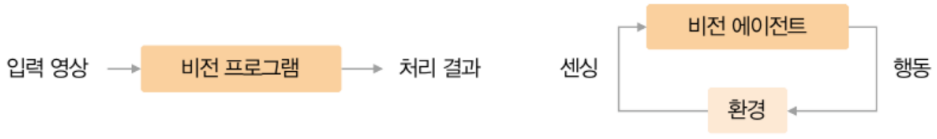


그림 6-2 환경과 상호작용하는 비전 에이전트

(a) 비전 프로그램: 환경과 상호작용 X

⇒ 비전 프로그램을 비전 에이전트로 확장하려면 적절한 그래픽 사용자 인터페이스(GUI)가 필수다.

(b) 비전 에이전트: 환경과 상호작용 O. (환경에 영향을 미치는 물리적 장치를 액추에이터라고 함)

6.2 PyQt를 이용한 사용자 인터페이스

PyQt 기초 프로그래밍

♦[6-1]PyQt로 간단한 GUI 만들기(버튼을 클릭하면 삑 소리 들려주기)

```
from PyQt5.QtWidgets import *
import sys
import winsound

class BeepSound(QMainWindow):
    def __init__(self) : #객체를 생성하면 자동으로 실행.
        super().__init__()
        self.setWindowTitle('삑 소리 내기')      # 윈도우 이름과 위치 지정
        self.setGeometry(200,200,500,100)

        shortBeepButton=QPushButton('짧게 삑',self)   # 버튼 생성
        longBeepButton=QPushButton('길게 삑',self)
        quitButton=QPushButton('나가기',self)
        self.label=QLabel('환영합니다!',self)
        # 멤버 변수로 만듦 -> 클래스 어디서든 접근 가능. 클래스로 생성한 객체에서도 접근 가능.

        shortBeepButton.setGeometry(10,10,100,30) # 버튼 위치와 크기 지정
        longBeepButton.setGeometry(110,10,100,30)
        quitButton.setGeometry(210,10,100,30)
        self.label.setGeometry(10,40,500,70)

        shortBeepButton.clicked.connect(self.shortBeepFunction) # 콜백 함수 지정
        longBeepButton.clicked.connect(self.longBeepFunction)
        quitButton.clicked.connect(self.quitFunction)

    def shortBeepFunction(self):
        self.label.setText('주파수 1000으로 0.5초 동안 삑 소리를 냅니다.')
        winsound.Beep(1000,500)

    def longBeepFunction(self):
        self.label.setText('주파수 1000으로 3초 동안 삑 소리를 냅니다.')
        winsound.Beep(1000,3000)

    def quitFunction(self):
        self.close()

app=QApplication(sys.argv)
win=BeepSound()
win.show()
app.exec_()
```

▼ 코드설명

1. BeepSound 클래스: GUI제작 지원
    - `QMainWindow`: 윈도우를 생성하고 관리하는 함수를 제공
    - 생성자 함수 __init__
        - 윈도우 이름과 위치 지정

          `self.setWindowTitle('삑 소리 내기')`: 윈도우 이름 지정

          `self.setGeometry(200,200,500,100)`: 창의 위치, 창의 크기 지정

    - 위젯 만들기

      `QPushButton('짧게 삑',self)`: 버튼

      `QLabel('환영합니다!',self)`: 레이블

    - 버튼 클릭 시 수행할 콜백 함수 지정.

      `shortBeepButton.clicked.connect(self.shortBeepFunction):`
      shortBeepButton 클릭 시 shortBeepFunction 함수 실행.

  - 실행할 함수들 정의

    - shortBeepFunction

      `self.label.setText()`: 레이블 위젯에 지정한 텍스트를 씀.

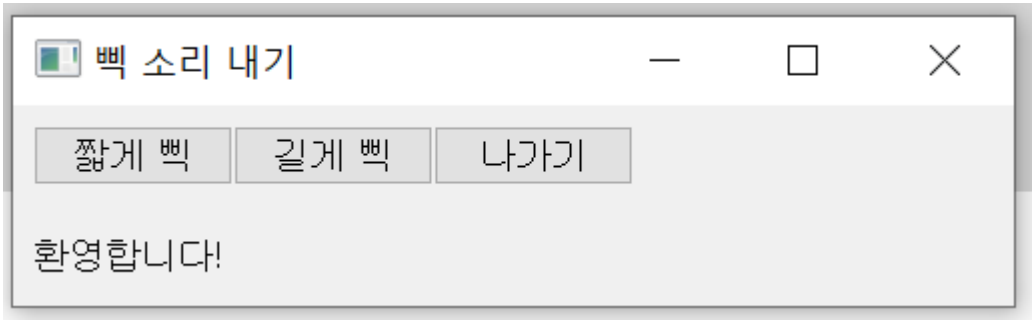
      `winsound.Beep(1000,500)`: 주파수 1000인 삑 소리를 500밀리초동안 들려준다.

2. `app=QApplication(sys.argv)`: PyQt 실행에 필요한 객체 app을 생성.
3. `win=BeepSound()`: BeepSound 클래스의 객체 생성. 생성자 함수 자동 실행됨.

⇒ 윈도우 생성하고, 위젯 4개 만들어지며 콜백 함수 등록됨.

4. `win.show()`: win에 해당하는 윈도우를 실제 화면에 나타냄.
5. `app.exec_()`: 무한 루프를 돌아 프로그램이 끝나는 것을 방지. 없으면 win을 화면에 띄우는 순간 프로그램 종료됨.

⇒ 종류는 콜백 함수로 한다.



OpenCV에 PyQt를 붙여 프로그램 확장하기

♦[6-2] OpenCV에 PyQt의 GUI 붙이기(비디오에서 프레임을 잡아 저장하기)

```
from PyQt5.QtWidgets import *
import sys
import cv2 as cv

class Video(QMainWindow):
```

```
    def __init__(self) :
        super().__init__()
        self.setWindowTitle('비디오에서 프레임 수집')  # 윈도우 이름과 위치 지정
        self.setGeometry(200,200,500,100)

        videoButton=QPushButton('비디오 켜기',self) # 버튼 생성
        captureButton=QPushButton('프레임 잡기',self)
        saveButton=QPushButton('프레임 저장',self)
        quitButton=QPushButton('나가기',self)

        videoButton.setGeometry(10,10,100,30)   # 버튼 위치와 크기 지정
        captureButton.setGeometry(110,10,100,30)
        saveButton.setGeometry(210,10,100,30)
        quitButton.setGeometry(310,10,100,30)

        videoButton.clicked.connect(self.videoFunction) # 콜백 함수 지정
        captureButton.clicked.connect(self.captureFunction)
        saveButton.clicked.connect(self.saveFunction)
        quitButton.clicked.connect(self.quitFunction)

    def videoFunction(self):
        self.cap=cv.VideoCapture(0,cv.CAP_DSHOW)  # 카메라와 연결 시도
        if not self.cap.isOpened(): self.close()

        while True:
            ret,self.frame=self.cap.read()
            if not ret: break
            cv.imshow('video display',self.frame)
            cv.waitKey(1)
    # frame과 cap은 멤버 변수로 선언

    def captureFunction(self):
        self.capturedFrame=self.frame
        cv.imshow('Captured Frame',self.capturedFrame)

    def saveFunction(self):       # 파일 저장
        fname=QFileDialog.getSaveFileName(self,'파일 저장','./')
        cv.imwrite(fname[0],self.capturedFrame)

    def quitFunction(self):
        self.cap.release()        # 카메라와 연결을 끊음
        cv.destroyAllWindows()
        self.close()

app=QApplication(sys.argv)
win=Video()
win.show()
app.exec_()
```

▼ 코드 설명

1. Video 클래스
    - 생성자 함수 __init__
        - 윈도우 이름과 위치,크기 지정

    - 위젯(버튼) 생성, 버튼의 위치 및 크기 지정
    - 콜백 함수 지정
- videoFunction 함수: <비디오 켜기>라는 videoButton의 콜백 함수

  1) `self.cap=cv.VideoCapture(0,cv.CAP_DSHOW)`: 웹 캠과 연결 시도

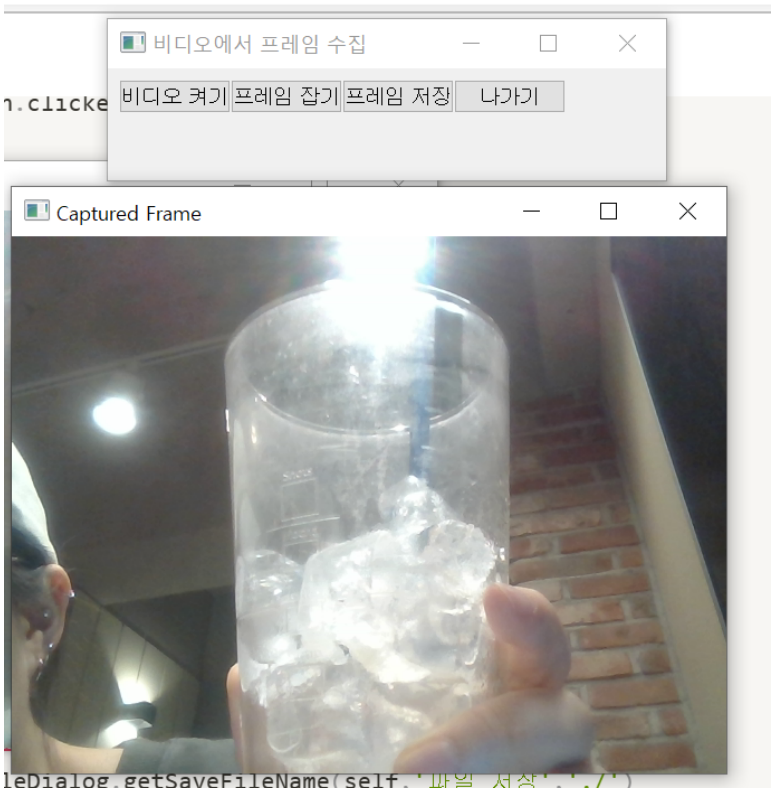
  2) if not: 연결이 안 됐을 때 오류 메시지 출력하고 프로그램 종료.

  3) 연결 성공 시 while loop 반복

    - `self.cap.read()`: 비디오에서 프레임 획득해 frame에 저장하고 윈도우에 표시
    - frame과 cap은 멤버 변수로 선언
        - cap은 quitFunction의 비디오 연결 끊을 때 사용
        - frame은 captureFunction에서 그 순간의 프레임을 capturedFrame으로 저장할 때 사용
- captureFunction: <프레임 잡기>라는 captureButton의 콜백 함수
    - 비디오 프레임을 저장한 frame을 capturedFrame 변수에 저장하고 디스플레이
- saveFunction: <프레임 저장> 이라는 saveButton의 콜백 함수
    - `QFileDialog.getSaveFileName(self,'브라우징 윈도우의 제목','./')`

      -3번째 인수: ‘./’는 현재 폴더에서 브라우징 하라

      -사용자가 입력한 파일의 이름을 반환
    - self.capturedFrame에 저장해둔 프레임을 사용자가 지정한 파일이름으로 저장. fname이 튜플이므로 fname[0]으로 지정한다.
- quitFunction: <나가기>라는 quitButton의 콜백 함수
    - 비디오 연결 끊음
    - OpenCV가 연 모든 윈도우 닫음.
    - 프로그램 종료.


비디오에서 프레임 수집
비디오 켜기
프레임 잡기
프레임 저장
나가기
Captured Frame