StreamingVideo

작성 중

StreamingVideo

```
import io
import picamera
import logging
import socketserver
from threading import Condition
from http import server
import os
from gpioControl import GpioControl
from listenSpeech import ListenSpeech
from urllib.parse import parse_qs


ONAIR = True
gpio = GpioControl()
listenspeech = ListenSpeech()

# Load HTML file
file = open("view.html", "r")
PAGE=file.read()
file.close()

class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()

    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            # New frame, copy the existing buffer's content and notify all
            # clients it's available
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                self.condition.notify_all()
            self.buffer.seek(0)
        return self.buffer.write(buf)

class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_POST(self):
        if self.path == '/msg':
```

```
            self.data_string = self.rfile.read(int(self.headers['Content-Length']))
            getMessage=self.data_string.decode()
            reString = listenspeech.speech(getMessage);
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            print("Return message = " + reString)
            self.wfile.write(reString.encode())
        elif self.path == '/forward':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goForward(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Forward ...")
        elif self.path == '/backward':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goBackward(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Backward ...")
        elif self.path == '/turnleftback':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goTurnleftback(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Turn Left back...")
        elif self.path == '/turnrightback':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
```

```
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goTurnrightback(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Turn Right back...")
        elif self.path == '/turnleftforward':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goTurnleftforward(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Turn Left forward...")
        elif self.path == '/turnrightforward':
            self.data_int = self.rfile.read(int(self.headers['Content-Length']))
            moveOrder=self.data_int
            self.send_response(200)
            if not moveOrder :
                reString = "Do not move"
            else :
                reString = gpio.goTurnrightforward(moveOrder)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Go Turn Right forward...")


    def do_GET(self):
        global ONAIR
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
```

```
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type', 'multipart/x-mixed-replace;
boundary=FRAME')
            self.end_headers()
            try:
                while ONAIR:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type', 'image/jpeg')
                    self.send_header('Content-Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
            except Exception as e:
                logging.warning(
                    'Removed streaming client %s: %s',
                    self.client_address, str(e))

        elif self.path == '/init':
            self.send_response(200)
            reString = "Camera position init : " + gpio.initMotorPosition()
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Init Cam ...")

        elif self.path == '/up':
            self.send_response(200)
            reString = gpio.moveUp()
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Move Up ...")

        elif self.path == '/down':
            self.send_response(200)
            reString = gpio.moveDown()
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
```

```
            self.wfile.write(reString.encode())
            print("Move Down ...")

        elif self.path == '/left':
            self.send_response(200)
            reString = gpio.moveLeft()
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Move Left ...")

        elif self.path == '/right':
            self.send_response(200)
            reString = gpio.moveRight()
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(reString.encode()))
            self.end_headers()
            self.wfile.write(reString.encode())
            print("Move Right ...")

        elif self.path == '/temperature':
            content = os.popen("vcgencmd measure_temp").readline()
            content = content.replace("temp=", "")
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content.encode()))
            self.end_headers()
            self.wfile.write(content.encode())

        else:
            self.send_error(404)
            self.end_headers()

class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True

with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    output = StreamingOutput()
    #Uncomment the next line to change your Pi's Camera rotation (in degrees)
    camera.rotation = 180
    camera.start_recording(output, format='mjpeg')
    try:
        address = ('', 8081)
        server = StreamingServer(address, StreamingHandler)
        server.serve_forever()
    finally:
        camera.stop_recording()
```

gpioControl.py

```
import RPi.GPIO as GPIO
from time import sleep


GPIO.setmode(GPIO.BOARD)

ina1 = 33
ina2 = 35
ena = 37

inb1 = 31
inb2 = 29
enb = 23

GPIO.setup(12, GPIO.OUT, initial=1)
GPIO.setup(18, GPIO.OUT, initial=1)
GPIO.setup(11, GPIO.OUT, initial=1) # light

GPIO.setup(ina1,GPIO.OUT)
GPIO.setup(ina2,GPIO.OUT)
GPIO.setup(ena,GPIO.OUT)

GPIO.setup(inb1,GPIO.OUT)
GPIO.setup(inb2,GPIO.OUT)
GPIO.setup(enb,GPIO.OUT)

p1 = GPIO.PWM(18, 50)  # 50 Hz
p2 = GPIO.PWM(12, 50)  # 50
p1.start(0)
p2.start(0)
p1.ChangeDutyCycle(0)
p2.ChangeDutyCycle(0)

pa=GPIO.PWM(ena,1000)
pa.start(25)

pb=GPIO.PWM(enb,1000)
pb.start(25)

verticalVal = 6.5
horizontalVal = 6

cameraPositionX = 6.5
cameraPositionY = 6

# Set up camera constants
IM_WIDTH = 640
IM_HEIGHT = 480
```

```
class GpioControl(object):

    def __init__(self):
        global verticalVal
        global horizontalVal
        global p1
        global p2
        p1.ChangeDutyCycle(6.5)
        p2.ChangeDutyCycle(6)
        sleep(0.1)
        p1.ChangeDutyCycle(0)
        p2.ChangeDutyCycle(0)
        verticalVal = 6.5
        horizontalVal = 6
        print("> Init Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))

    def click(self):
        GPIO.output(11, GPIO.LOW)
        sleep(0.5)
        GPIO.output(11, GPIO.HIGH)
        sleep(1)

    def __del__(self):
        global p1
        global p2
        p1.stop()
        p2.stop()
        print(" GPIO.__del__() ")
        GPIO.cleanup()

    def cleanUp(self):
        global p1
        global p2
        p1.stop()
        p2.stop()
        print(" GPIO.cleanUp() ")
        GPIO.cleanup()
        sleep(2)

    def initMotorPosition(self):
        # Init
        global verticalVal
        global horizontalVal
        global p1
        global p2
        p1.ChangeDutyCycle(6.5)
        p2.ChangeDutyCycle(6)
        sleep(0.1)
```

```
        p1.ChangeDutyCycle(0)
        p2.ChangeDutyCycle(0)
        verticalVal = 6.5
        horizontalVal = 6
        print("> Init Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))
        return "Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal)

    def moveUp(self):
        global verticalVal
        global horizontalVal
        global p2
        verticalVal = round(verticalVal-0.2, 1)
        p2.ChangeDutyCycle(verticalVal)
        print("> UP Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))
        sleep(0.1)
        p2.ChangeDutyCycle(0)
        return "Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal)

    def moveDown(self):
        global verticalVal
        global horizontalVal
        global p2
        verticalVal = round(verticalVal+0.2, 1)
        p2.ChangeDutyCycle(verticalVal)
        print("> Down Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))
        sleep(0.1)
        p2.ChangeDutyCycle(0)
        return "Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal)

    def moveRight(self):
        global verticalVal
        global horizontalVal
        global p1
        horizontalVal = round(horizontalVal-0.2, 1)
        p1.ChangeDutyCycle(horizontalVal)
        print("> Right Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))
        sleep(0.1)
        p1.ChangeDutyCycle(0)
        return "Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal)

    def moveLeft(self):
        global verticalVal
        global horizontalVal
        global p1
        horizontalVal = round(horizontalVal+0.2, 1)
        p1.ChangeDutyCycle(horizontalVal)
        print("> Left Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal))
        sleep(0.1)
        p1.ChangeDutyCycle(0)
        return "Vert=" + str(verticalVal) + ",Hort=" + str(horizontalVal)
```

```
def goForward(self,time):
    print("Go forward")
    pa.ChangeDutyCycle(75)
    GPIO.output(ina1,GPIO.HIGH)
    GPIO.output(ina2,GPIO.LOW)
    pb.ChangeDutyCycle(75)
    GPIO.output(inb1,GPIO.HIGH)
    GPIO.output(inb2,GPIO.LOW)
    sleep(0.1*int(time))
    GPIO.output(ina1,GPIO.LOW)
    GPIO.output(ina2,GPIO.LOW)
    GPIO.output(inb1,GPIO.LOW)
    GPIO.output(inb2,GPIO.LOW)
    return " >> Go Forward"

def goBackward(self,time):
    print("Go backward")
    pa.ChangeDutyCycle(75)
    GPIO.output(ina1,GPIO.LOW)
    GPIO.output(ina2,GPIO.HIGH)
    pb.ChangeDutyCycle(75)
    GPIO.output(inb1,GPIO.LOW)
    GPIO.output(inb2,GPIO.HIGH)
    sleep(0.1*int(time))
    GPIO.output(ina1,GPIO.LOW)
    GPIO.output(ina2,GPIO.LOW)
    GPIO.output(inb1,GPIO.LOW)
    GPIO.output(inb2,GPIO.LOW)
    return " >> Go Backward"

def goTurnleftback(self,time):
    print("Go Turn Left back")
    pa.ChangeDutyCycle(75)
    GPIO.output(inb1,GPIO.LOW)
    GPIO.output(inb2,GPIO.HIGH)
    sleep(0.1*int(time))
    GPIO.output(ina1,GPIO.LOW)
    GPIO.output(ina2,GPIO.LOW)
    GPIO.output(inb1,GPIO.LOW)
    GPIO.output(inb2,GPIO.LOW)
    return " >> Go Turn Left back"

def goTurnrightback(self,time):
    print("Go Turn Right back")
    pb.ChangeDutyCycle(75)
    GPIO.output(ina1,GPIO.LOW)
    GPIO.output(ina2,GPIO.HIGH)
    sleep(0.1*int(time))
```

```
        GPIO.output(ina1,GPIO.LOW)
        GPIO.output(ina2,GPIO.LOW)
        GPIO.output(inb1,GPIO.LOW)
        GPIO.output(inb2,GPIO.LOW)
        return " >> Go Turn Right back"

    def goTurnleftforward(self,time):
        print("Go Turn Left forward")
        pa.ChangeDutyCycle(75)
        GPIO.output(inb1,GPIO.HIGH)
        GPIO.output(inb2,GPIO.LOW)
        sleep(0.1*int(time))
        GPIO.output(ina1,GPIO.LOW)
        GPIO.output(ina2,GPIO.LOW)
        GPIO.output(inb1,GPIO.LOW)
        GPIO.output(inb2,GPIO.LOW)
        return " >> Go Turn Left forward"

    def goTurnrightforward(self,time):
        print("Go Turn Right forward")
        pb.ChangeDutyCycle(75)
        GPIO.output(ina1,GPIO.HIGH)
        GPIO.output(ina2,GPIO.LOW)
        sleep(0.1*int(time))
        GPIO.output(ina1,GPIO.LOW)
        GPIO.output(ina2,GPIO.LOW)
        GPIO.output(inb1,GPIO.LOW)
        GPIO.output(inb2,GPIO.LOW)
        return " >> Go Turn Right forward"




    def move_to_position(self,ObjX, ObjY):
        global cameraPositionX
        global cameraPositionY
        global p1
        global p2
        print(" >> Move to location x="+str(ObjX)+", y="+str(ObjY))
        moveLoop = True
        movX = int(IM_WIDTH/2)-ObjX
        movY = int(IM_HEIGHT/2)-ObjY
        print(" >> Center location movX="+str(movX)+", movY="+str(movY))

        xx = 1
        xy = 0
        yx = 1
        yy = 0
        while(moveLoop):
            if( xy < abs(movX) ):
```

```
                p1.ChangeDutyCycle(cameraPositionX)
                sleep(0.1)
                p1.ChangeDutyCycle(0)
                print("xx=" + str(xx) + ", xy="+ str(xy) +" cameraPositionX="
+str(round(cameraPositionX,1)))
                xy = xx*xx * 12
                xx = xx + 1
                if(movX > 0): cameraPositionX = cameraPositionX - 0.2
                else: cameraPositionX = cameraPositionX + 0.2
            if( yy < abs(movY) ):
                p2.ChangeDutyCycle(cameraPositionY)
                sleep(0.1)
                p2.ChangeDutyCycle(0)
                print("yx=" + str(yx) + ", yy="+ str(yy) +" cameraPositionY="
+str(round(cameraPositionY,1)))
                yy = yx*yx * 12
                yx = yx + 1
                if(movY > 0): cameraPositionY = cameraPositionY + 0.2
                else: cameraPositionY = cameraPositionY - 0.2
            elif( xy >= movX and yy >= movY):
                print(" >> Center location movX="+str(movX)+", movY=" + str(movY))
                print(" >> Position location xy="+str(xy)
                      +", yy="+str(yy)
                      +" cameraPositionX="+str(round(cameraPositionX,1))+
                       " cameraPositionY="+str(round(cameraPositionY,1))+" .. ")
                moveLoop = False
```

ListenSpeech

```
from googletrans import Translator
from google_speech import Speech
from time import sleep

translator = Translator()

class ListenSpeech(object):
    def speech(self,message):
        ko_result = translator.translate(message, dest='ko')
        print(' -> ', ko_result.text)
        speech = Speech(ko_result.text, 'ko')
        speech.play()
        en_result = translator.translate(message, dest='en')
        speech = Speech(en_result.text, 'en')
        speech.play()
        it_result = translator.translate(message, dest='it')
        speech = Speech(it_result.text, 'it')
        speech.play()
        ja_result = translator.translate(message, dest='ja')
        speech = Speech(ja_result.text, 'ja')
        speech.play()
```

```
        returnString = ko_result.text + " , EN=" + en_result.text + " , IT=" +
it_result.text + ", JA=" + ja_result.text
        print("Return message = " + returnString)
        return returnString
```

view.html

```
<html>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta charset="UTF-8" />
<head>
<title>Raspberry Pi - HYUNSU Camera</title>

<script type="text/javascript">

window.onload = function() {
    document.getElementById("btnInit").onclick = getInitFunc;
    document.getElementById("btnMsg").onclick = getMsgFunc;

    document.getElementById("btnLeft").onclick = getLeftFunc;
    document.getElementById("btnRight").onclick = getRightFunc;
    document.getElementById("btnUp").onclick = getUpFunc;
    document.getElementById("btnDown").onclick = getDownFunc;

    document.getElementById("btnForward").onclick = getForwardFunc;
    document.getElementById("btnBackward").onclick = getBackwardFunc;
    document.getElementById("btnTurnrightback").onclick = getTurnrightbackFunc;
    document.getElementById("btnTurnleftback").onclick = getTurnleftbackFunc;
    document.getElementById("btnTurnrightforward").onclick = getTurnrightforwardFunc;
    document.getElementById("btnTurnleftforward").onclick = getTurnleftforwardFunc;
}

var xhr;

function getInitFunc() {
    var fName = "/init";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
}
```

```
function getMsgFunc() {
    var fName = "/msg";
    // var data = new FormData();
    data=document.getElementById("textInput").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}


function getLeftFunc() {
    var fName = "/left";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
}

function getRightFunc() {
    var fName = "/right";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
```

```
}

function getUpFunc() {
    var fName = "/up";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
}

function getDownFunc() {
    var fName = "/down";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
}



function getForwardFunc() {
    var fName = "/forward";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
```

```
    xhr.send(data);
}


function getBackwardFunc() {
    var fName = "/backward";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}


function getTurnleftbackFunc() {
    var fName = "/turnleftback";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}


function getTurnrightbackFunc() {
    var fName = "/turnrightback";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
```

```
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}


function getTurnleftforwardFunc() {
    var fName = "/turnleftforward";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}


function getTurnrightforwardFunc() {
    var fName = "/turnrightforward";
    data=document.getElementById("moveCnt").value;
    xhr = new XMLHttpRequest();
    xhr.open("post", fName, true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("systemMessage").innerHTML = xhr.responseText;
            } else {
                alert("ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(data);
}



function getTemperatureFunc() {
    var fName = "/temperature";
    xhr = new XMLHttpRequest();
    xhr.open("get", fName, true);
    xhr.onreadystatechange = function() {
```

```
        if (xhr.readyState == 4) {
            if (xhr.status == 200) {
                document.getElementById("temperature").innerHTML = xhr.responseText;
            } else {
                alert(" temperature ERROR : " + xhr.status);
            }
        }
    }
    xhr.send(null);
}

var timerId = null;
timerId = setInterval(getTemperatureFunc, 30000);
//clearInterval(timerId);

</script>

</head>
<body>
<form name="frm">
<center>
<h3>꼬꼬락 KKoLack</h3>
<h4>system temperature = <span id="temperature">...</span></h4>

<table border=0>
<tr>
    <!-- Image section -->
    <td>
        <img src="stream.mjpg" width="300">
    </td>
    <!-- Control section -->
    <td>
        <table border=0>
            <tr>
                <td>
                Message = <span id="systemMessage"> ... </span>
                <br>
                <input type=text id="textInput" lenght=20 ><input type="button"
value="Send Message" id="btnMsg" />
                </td>
            </tr>
            <tr>
                <td>
                    <center>
                    Control Camera Section

                    <table>
                    <tr>
                        <td></td>
                        <td><input type="button" value=" ^ " id="btnUp" /></td>
```

```
                    <td></td>
                </tr>
                <tr>
                    <td><input type="button" value=" < " id="btnLeft" /></td>
                    <td></td>
                    <td><input type="button" value=" > " id="btnRight" /></td>
                </tr>
                <tr>
                    <td></td>
                    <td><input type="button" value=" v " id="btnDown" /></td>
                    <td></td>
                </tr>
                </table>
                </center>
            </td>
        </tr>
        <tr>
            <td>
                <center>
                Control Engine Section
                <br>Move cont (0.1 ~ 10) = <input type=text id="moveCnt"
maxlength="5" size="5" />
                <br>
                <table>
                <tr>
                    <td></td>
                    <td><input type="button" value=" ^ " id="btnForward" /></td>
                    <td></td>
                </tr>
                <tr>
                    <td><input type="button" value=" ^ " id="btnTurnleftforward" /
></td>
                    <td></td>
                    <td><input type="button" value=" ^ "
id="btnTurnrightforward" /></td>
                </tr>
                <tr>
                    <td><input type="button" value=" v " id="btnTurnleftback" /></
td>
                    <td></td>
                    <td><input type="button" value=" v " id="btnTurnrightback" /
></td>
                </tr>
                <tr>
                    <td></td>
                    <td><input type="button" value=" v " id="btnBackward" /></td>
                    <td></td>
                </tr>
                </table>
                </center>
```

```
                </td>
            </tr>
        </table>
    </td>
</tr>
</table>
    <input type="button" value="Initialization Camera" id="btnInit" /><br>
</center>
</form>
</body>
</html>
```

http://web.joang.com:9000/jcook/StreamingVideo

Revision #4
★Created 2023-06-01 14:48:58 UTC by Hyeon Su Ryu
Updated 2023-06-02 12:36:25 UTC by Hyeon Su Ryu