2025년 1학기 데이터통신 4주차 과제 보고서

- 유니코드 → 모스부호 통신 -

202001156 정보통계학과 김수영

과제 해결

영상링크

https://www.youtube.com/watch?v=qyaCkdLKkgU

요구사항

- Timing unit: 100ms (o.1s)
- Frequency: 523.251hz
- Channels: 1
- Sample rate: 48000
- 모스부호신호는 국제 표준 적용

송신기능

```
12  def send_data(text):
13      unicode = uni.str2uni(text)
14      audio = morse.morse2audio(morse.text2morse(unicode))
15
16      p = pyaudio.PyAudio()
17
18      stream = p.open(format=pyaudio.paInt16,
19                      channels=1,
20                      rate = 48000,
21                      output= True)
22
23      chunk_size = 1024
24
25      for i in range(0, len(audio), chunk_size):
26          chunk = audio[i:i+chunk_size]
27          stream.write(struct.pack('<'+('h'*len(chunk)),*chunk))
28
29      stream.stop_stream()
30      stream.close()
31      p.terminate()
```

요구사항에 맞춰서 작성한 송신함수이다. 함수에 사용된 morse 모듈은 이전 과제에 사용한 모스 부호 변환 코드이다. morse2audio와 text2morse는 각각 글자를 모스부호로, 모스부호를 오디오 리스트로 변환하는 코드이다. 또한, 처음에 입력받은 텍스트를 유니코드로 변환하는 부분을 추가하였다.

```python
def morse2audio(morse):
    INTMAX = 32767
    t = 0.1
    fs = 48000
    f = 523.251
    audio = []
    for m in morse:
        if m == '.':
            for i in range(int(t*fs*1)):
                audio.append(int(INTMAX*math.sin(2*math.pi*f*(i/fs))))
            for i in range(int(t*fs*1)):
                audio.append(int(0))
        elif m == '-':
            for i in range(int(t*fs*3)):
                audio.append(int(INTMAX*math.sin(2*math.pi*f*(i/fs))))
            for i in range(int(t*fs*1)):
                audio.append(int(0)) # dot와 dash 사이의 1unit 추가
        elif m == '/':
            for i in range(int(t*fs*2)):
                audio.append(int(0)) # 단어와 단어 사이 공백, 문자와 문자 사
        elif m == ' ':
            for i in range(int(t*fs*2)):
                audio.append(int(0)) # 문자와 문자 사이, 문자와 문자 사이는

    return audio
```

morse2audio 함수의 경우 요구사항에 맞춰 잘 frequency와 rate, unit(코드에서 t로 정의된 것)을 정의하였으며 스피커와 마이크의 스펙 한계로 인해 pyaudio의 포맷은 paint16으로 설정하였다. 따라서 2바이트를 정보로 갖는 포맷의 특성상 INTMAX 역시 2^15 -1 인 32767로 설정하였다. 모스부호 송신 간격 역시 국제 규정에 맞게 설정하였다.

```python
def text2morse(text):
    text = text.upper()
    morse = ''

    for t in text:
        if t == ' ':
            morse = morse + '/ '
        for key, value in code.items():
            if t == key:
                morse = morse + value + " "
    return morse
```

text2morse 함수는 단순히 텍스트를 모스부호 양식으로 바꿔주는 함수이다. 새로 추가한 모스부호 규칙으로 바꾸었다. (code.item 부분)

수신 기능

```python
def audio2morse():
    m = ''
    p = pyaudio.PyAudio()
    threshold = 220
    stream = p.open(format = pyaudio.paInt16,
                    channels = 1,
                    rate = 48000,
                    input = True
                    )

    unit_samples = int(0.1 * 48000)

    # 목표 2
    while True:
        data = stream.read(unit_samples,exception_on_overflow=False)
        samples = struct.unpack('<' + ('h' * unit_samples), data)
        if statistics.mean([abs(s) for s in samples]) > threshold:
            m += '.'
            print(m)
            break

    seen = 0
    while True:
        data = stream.read(unit_samples, exception_on_overflow=False)
        samples = struct.unpack('<' + ('h' * unit_samples), data)
        mean = statistics.mean([abs(s) for s in samples])
        # 목표 1
        if mean > threshold:
            m += '.'
            seen = 0
        else:
            m += ' '
            seen += 1

        print("Signal: ", m)

        # 목표 3
        if seen >= 35:
            break

    stream.stop_stream()
    stream.close()
    p.terminate()

    print("Input Signal: " + m)

    m = m.replace('...', '-')
    return m
```

마이크로 모스부호 신호를 받으면 이를 문자열로 인식하는 함수이다. 수신 기능에서 제일 중요한 함수로 목

표1, 2, 3을 해결한 부분을 표시하였다. 해당 함수의 알고리즘을 설명하자면

목표2 - 동기화

신호의 시작점을 인식하는 부분이다. 스트림을 통해 계속 마이크 신호를 받다가 이 값이 특정 threshold값을 넘으면 해당 신호를 모스부호의 시작 부분으로 인식한다. 그 전까지는 계속 대기 상태에 있는다.

목표1 - 소음 처리

소음을 처리하는 부분이다. Statistics 라이브러리의 평균 함수를 이용하여 평균값이 threshold 이상이면 신호 아니면 소음으로 처리하게 하였다. 마찬가지로 동기화 부분에서도 이를 활용하여 threshold가 넘지 않는 신호들은 모조리 소음으로 처리하여 신호를 받지 않게 설정하였다.

threshold의 값은 여러 실험을 통해 내린 값이며 영상에서 확인할 수 있듯이 적절한 거리에서 스피커를 들고 있는 상황에서의 값이다.

추가적으로 신호 처리 시 평균을 이용한 것은 '-' 신호를 인식하기 위함이다. 이 함수의 목적은 모스부호를 '.' 과 공백으로 우선 인식하는 것이다. 따라서 '-' 신호는 '…'으로 변환되어야 하는데 표준편차의 경우에는 연속적으로 높은 신호가 들어올 시 값의 변화가 없어서 오히려 작아지므로 '-' 신호를 감지할 수 없으며 최댓값은 갑작스러운 소음에 민감하므로 이를 안정화할 수 있는 평균을 신호 처리 값으로 선정하였다. 추가적으로 신호는 사인파의 형상을 띄기에 어느 신호건 간에 평균 값은 항상 0에 근접하게 나올 것이다. 따라서 신호의 절댓값의 평균을 사용하는 것으로 이를 해결하였다.

목표3 - 신호의 끝

신호의 끝을 정의하는 부분이다. 신호가 들어오면 seen의 값을 초기화하고 그렇지 않을 경우 seen의 값을 1 늘린다. Unit time이 0.1초이므로 한 번의 while문을 돌때마다 0.1초 간격으로 seen의 값이 업데이트 되며 따라서 seen의 값이 35이상 즉, 3.5초가 넘었는데도 신호가 입력되지 않을 경우 신호가 더 이상 없다고 판단하여 함수를 종료하며 모스부호 신호의 입력을 중단한다.

목표를 전부 해결한 후 '…'은 '-'으로 바꿔주면 된다.

해당 부분은 이전 과제와 완전히 동일하다.

```
82  def receive_data():
83      signal = morse.morse2text(morse.tab2slash(morse.unit2text(audio2morse())))
84      return uni.uni2str(signal)
```

이후 받은 신호를 morse에 있는 모스부호 변환 함수를 이용해 다시 텍스트로 복호화한다. 이때, 모스부호로 받은 신호는 유니코드이므로 이를 다시 원래 텍스트로 복호화하는 부분을 추가한다.

```
78  def unit2text(input):
79      output = ""
80      for i in range(0,len(input),2):
81          if input[i] == ".":
82              output = output + "."
83          elif input[i] == "-":
84              output = output + "-"
85          elif input[i] == " ":
86              output = output + " "
87      return output
88
89  def tab2slash(input):
90      input_list = list(input)
91      for i in range(0, len(input_list)-1):
92          if input_list[i] == " " and input_list[i+1] == " ":
93              input_list[i+1] = "/"
94      return "".join(input_list)
95
96  def convert(word):
97      for key, val in code.items():
98          if val == word:
99              return key
100     for key, val in code.items():
101         if val == word:
102             return key
103
104 def morse2text(morse):
105     letter = ""
106     text = ""
107
108     for t in morse:
109         if t != " ":
110             if t == "/":
111                 text = text + " "
112             else:
113                 letter = letter + t
114         else:
115             if letter:
116                 text = text + convert(letter)
117                 letter = ""
118
119     if letter:
120         text = text + convert(letter)
121
122     return text
```

사용된 함수는 이전 과제에 사용한 함수와 동일하며 이 함수의 동작 방식 및 검증은 이전 과제에서 다루었으므로 구체적인 설명은 생략한다. 또한, convert 함수가 기존 모스부호 규칙이 아닌 이번 과제에서 설정한 dictionary를 사용하는 것으로 바꾸어 주었다.

```python
def str2uni(text):
    # str=>hex
    byte_hex = text.encode('utf-8')
    byte_string = byte_hex.hex().upper()
    print(f'Byte String: {byte_string}')
    return byte_string

def uni2str(byte_string):
    # byte_string => str
    client_byte_hex = bytes.fromhex(byte_string)
    client_byte_string = client_byte_hex.hex().upper()
    print(f'Client Byte String: {client_byte_string}')
    client_output = client_byte_hex.decode('utf-8')
    return client_output

```

마지막으로 텍스트와 유니코드를 변환하는 코드이다. unicode_conv.py라는 파일에 작성되어 있다.

결론

최종적으로 일반적인 텍스트를 유니코드로 변환하고 변환된 유니코드를 모스신호로 송신하여 마이크와 스피커를 통해 통신할 수 있는 코드를 작성하였으며 정상적으로 작동함을 영상을 통해 확인할 수 있다.

영상링크

https://www.youtube.com/watch?v=qyaCkdLKkgU