



Dipartimento di Scienze e Tecnologie
Corso di laurea in Informatica Applicata

Smart Raspberry Pi Cat-camera

Studente: Domenico Scognamiglio
Matricola: 0120/000169

Introduzione

- Obiettivo: realizzare tramite Raspberry Pi una smart camera che consenta il monitoraggio del proprio gatto anche quando si è lontani da casa
- Molto popolari e acquistabili tipicamente a prezzi elevati



Furbo - VIDEOCAMERA per Cani: Telecamera HD WiFi per Animali con Audio Bidirezionale, Visione Notturna, Alerta Latrato e Lancio di Croccantini
di [Furbo](#)
 9.317 voti | 59 domande con risposta

Prezzo: **199,00 €** ✓prime
Tutti i prezzi includono l'IVA.



SKYMEE Dog Camera Videocamera per Cani con Treat Dispenser Lancio Croccantini Videocamera per Animali Wifi Telecomando 1080P HD Visione Notturna Audio Bidirezionale, Compatibile con Alexa
di [SKYMEE](#)
 85 voti

Prezzo: **139,99 €** ✓prime
Tutti i prezzi includono l'IVA.

Telegram Bot

- Per il controllo semplificato della smart camera e' stato sviluppato un Bot per Telegram, un'app di messaggistica gratuita multiplataforma tra le più utilizzate al mondo.

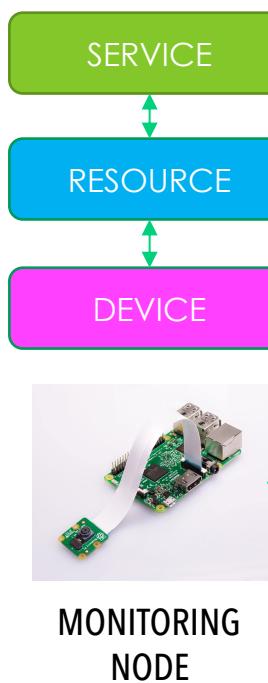
Bots: An introduction for developers



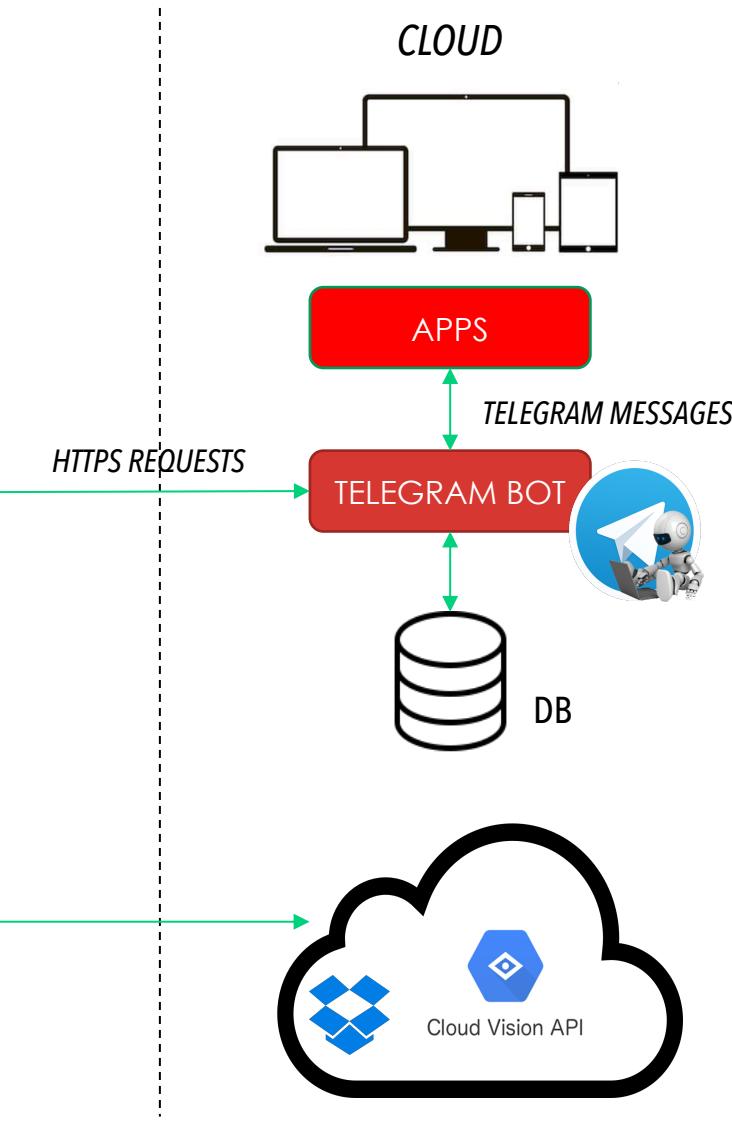
Bots are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and [inline requests](#). You control your bots using HTTPS requests to our [Bot API](#).

- Una volta avviato il Bot, l'utente tramite Telegram sarà in grado di interagire con la smart camera richiedendo i servizi messi a disposizione e ricevendo notifiche sugli eventi di interesse

LOCAL



CLOUD



Librerie utilizzate

- ❑ *python-telegram-bot*: interfaccia Python per le Telegram Bot API



<https://github.com/python-telegram-bot/python-telegram-bot>



python-telegram-bot

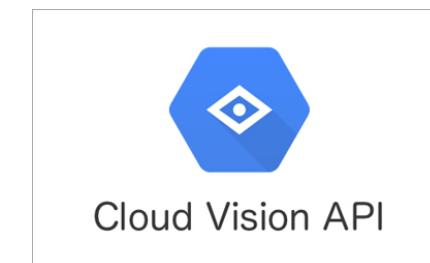
- ❑ *Motion*: per la rilevazione del movimento tramite Raspberry Camera

<https://motion-project.github.io>



- ❑ *Google Cloud Vision API*: per verificare se il gatto è presente o meno nelle foto

<https://cloud.google.com/vision/docs>



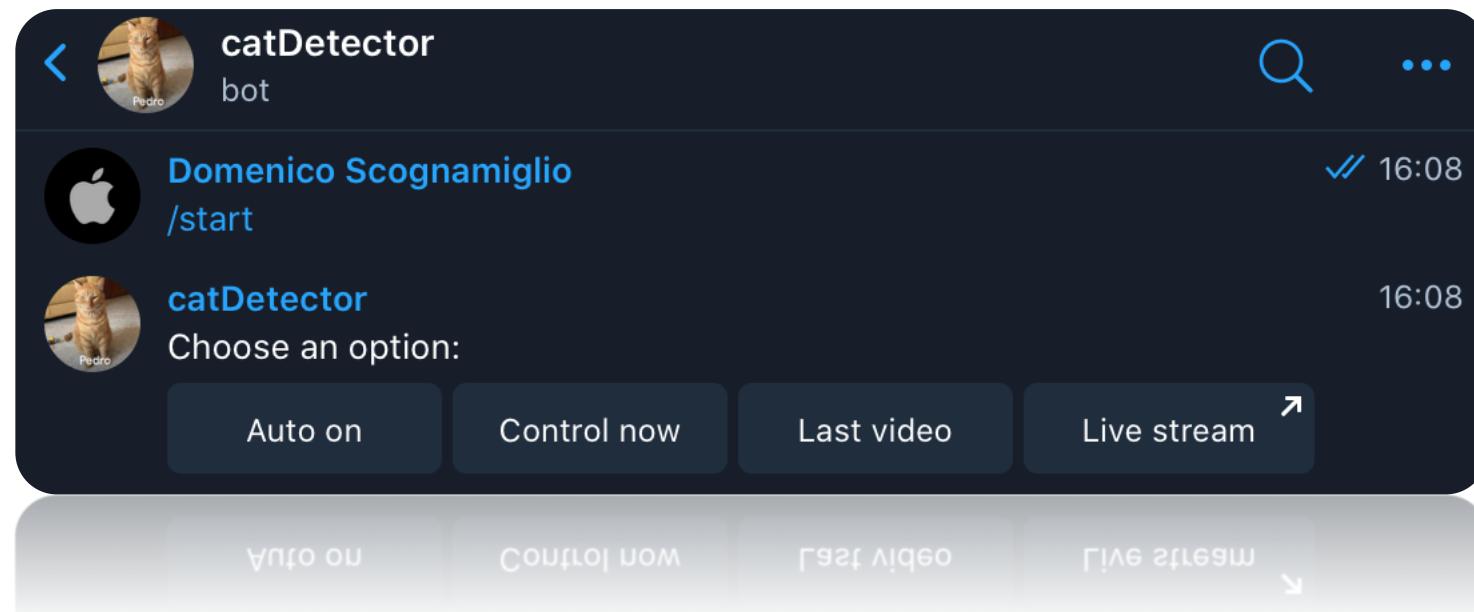
- ❑ *LifxLAN*: per il controllo tramite rete locale di una lampadina smart



<https://github.com/mclarkk/lifxlan>



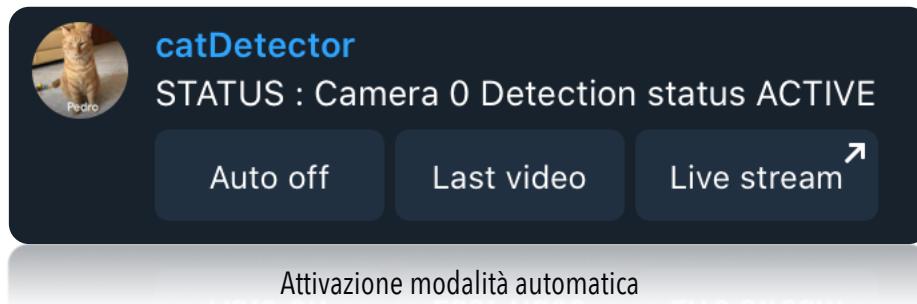
Avvio Bot



- Una volta avviato il Bot con il comando */start* viene mostrata una rapida descrizione delle funzionalità disponibili.
- Auto on:** attivazione della modalità automatica
- Control now:** verifica istantanea di presenza gatto
- Last video:** richiesta dell'ultimo video registrato (se presente)
- Live stream:** live streaming

Modalità automatica

- *Obiettivo:* rilevazione automatica del movimento tramite Raspberry Camera e ricezione di notifica nel caso in cui il movimento sia stato generato dal gatto (si suppone che possano esistere altre fonti di movimento alle quali non siamo interessati es., altre persone presenti in casa).
- All'attivazione della modalità automatica da parte dell'utente, corrisponde una richiesta HTTP di tipo GET per l'abilitazione di Motion. Ogni qualvolta verrà rilevato un movimento Motion salverà un video ed una foto significativa del movimento rilevato.



Attivazione modalità automatica

```
def setAutoModeOn():
    now,sunset,sunrise=get_times()
    if now>sunset:
        light_on()
        schedule_light_on(cron,sunset)
        schedule_light_off(cron,sunrise)
        print("auto mode on")
        requests.get('http://localhost:8080/0/detection/start')
    status=requests.get('http://localhost:8080/0/detection/status')
    return status.text
```

- E' possibile configurare Motion per l'invocazione automatica di uno script da eseguire al momento in cui si conclude la rilevazione di un movimento (*on_picture_save* in *motion.conf* file)

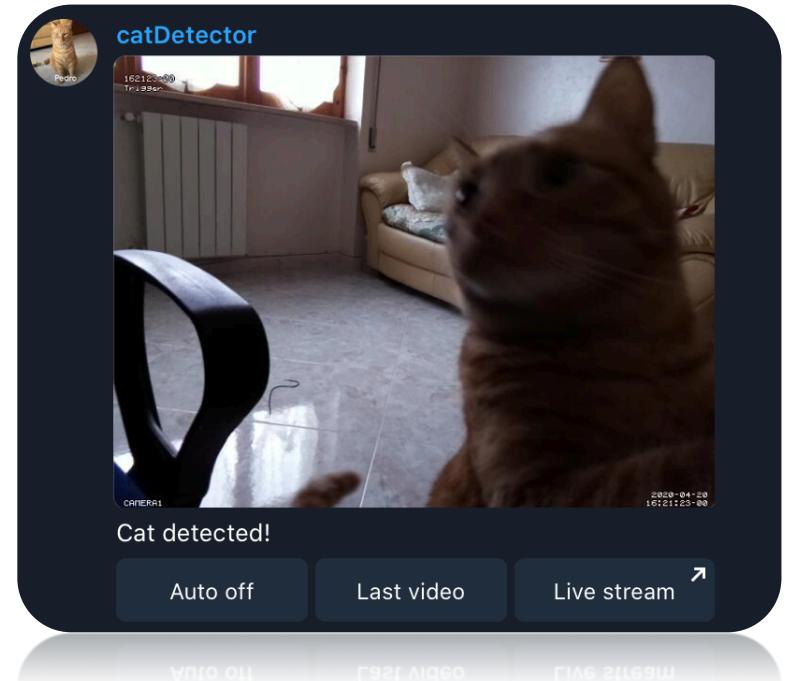
on_picture_save script(1)

- Al salvataggio di un'immagine da parte di Motion, viene eseguito uno script che innanzitutto verifica, tramite le Google Cloud Vision API, se nell'immagine è presente un gatto.

```
#Utilizzo delle Google Cloud Vision API per verificare se nell'immagine e' presente un gatto
def checkCat(image):
    client = vision.ImageAnnotatorClient()
    content = image.read()
    image = types.Image(content=content)
    response = client.label_detection(image=image)
    labels = response.label_annotations
    for label in labels:
        print("LABEL:", label.description, " SCORE:", label.score)
        if label.description == 'Cat' and label.score > 0.8:
            return True
    return False
```

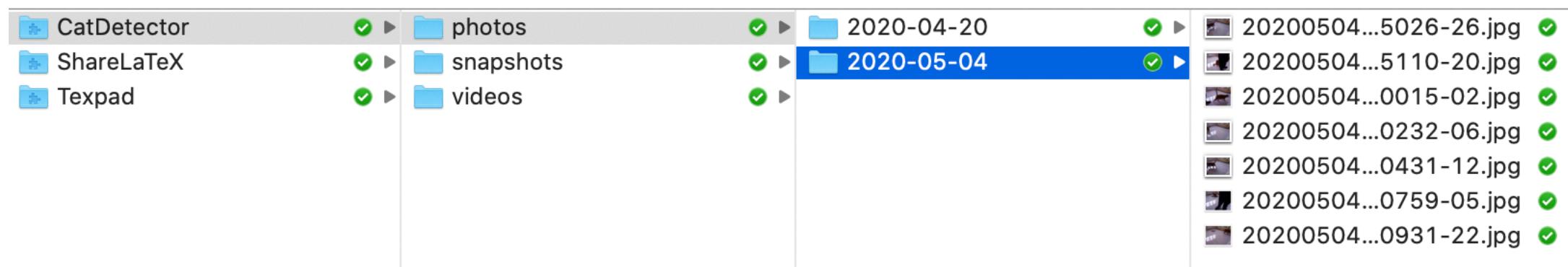
- In caso affermativo viene inviato un messaggio all'utente, con inclusa una foto, che lo avvisa dell'avvenuta identificazione.

```
last_video_path = get_last_motion_video_path()
if cat:
    bot.send_chat_action(chat_id=CHAT_ID, action=ChatAction.UPLOAD_PHOTO)
    bot.send_photo(chat_id=CHAT_ID, photo=last_image, caption="Cat detected!", reply_markup=get_Keyboard())
    last_image.seek(0)
    upload_to_dropbox(last_image, os.path.basename(last_image_path), type='/photos/')
    upload_to_dropbox(open(last_video_path, 'rb'), os.path.basename(last_video_path), type='/videos/')
    if os.path.exists(MOTION_FOLDER+'cat.mp4'):
        os.remove(MOTION_FOLDER+'cat.mp4')
    os.rename(last_video_path, os.path.dirname(last_video_path)+'/cat.mp4')
else:
    print("no cat detected in auto mode")
    os.remove(last_video_path)
os.remove(last_image_path)
```



on_picture_save script (2)

- In caso di cat-hit, vengono archiviate sul cloud (Dropbox) la foto e il video corrispondenti in delle sottocartelle denominate con la data corrente. A causa della ridotta memoria locale, viene mantenuto localmente solo il video più recente in cui sia presente un gatto (in quanto potrebbe essere richiesto dall'utente)



Archiviazione su Dropbox di foto e video

- Per il monitoraggio automatico anche in scarse condizioni di luminosità vengono schedulati tramite cron e la libreria LifxLAN l'accensione e lo spegnimento di una lampadina Wi-Fi Lifx, rispettivamente agli orari di tramonto e alba.

Altre opzioni (1)

- **Last video:** l'utente, magari a seguito della ricezione di una notifica di avvistamento gatto, potrebbe voler vedere il video dell'evento. Tramite l'opzione Last video viene recuperato il video più recente e inviato all'utente.

```
def get_last_cat_video(update,context):
    if os.path.exists(MOTION_FOLDER + '/cat.mp4'):
        last_video=open(MOTION_FOLDER + '/cat.mp4','rb')
        context.bot.send_chat_action(chat_id=update.effective_chat.id, action=ChatAction.UPLOAD_VIDEO)
        context.bot.send_video(chat_id=update.effective_chat.id,video=last_video,caption="Last cat video",reply_markup=get_Keyboard())
    else:
        context.bot.send_message(chat_id=update.effective_chat.id, text='No cat video found',reply_markup=get_Keyboard())
```

- **Auto off:** disabilitazione modalità automatica. Viene inviata una richiesta HTTP GET a Motion per metterlo in pausa e vengono cancellati i task schedulati di cron per l'accensione e lo spegnimento della lampadina.

```
def setAutoModeOff():
    light_off()
    cron.remove_all()
    cron.write()
    print("auto mode off")
    requests.get('http://localhost:8080/0/detection/pause')
    status = requests.get('http://localhost:8080/0/detection/status')
    return status.text
```

Altre opzioni (2)

- Control now: se la modalità automatica è disattivata, l'utente potrebbe voler sapere se in un determinato istante il gatto si trovi o meno nel range visibile della camera, avviando magari un live stream in caso affermativo. Inviando la richiesta GET mostrata Motion salverà uno snapshot.

```
def capture_snapshot():
    now, sunset, sunrise = get_times()
    if now > sunset:
        status=get_light_status()
        if status==0:
            light_on()
        requests.get('http://localhost:8080/0/action/snapshot')
    if now > sunset and status==0:
        light_off()
```

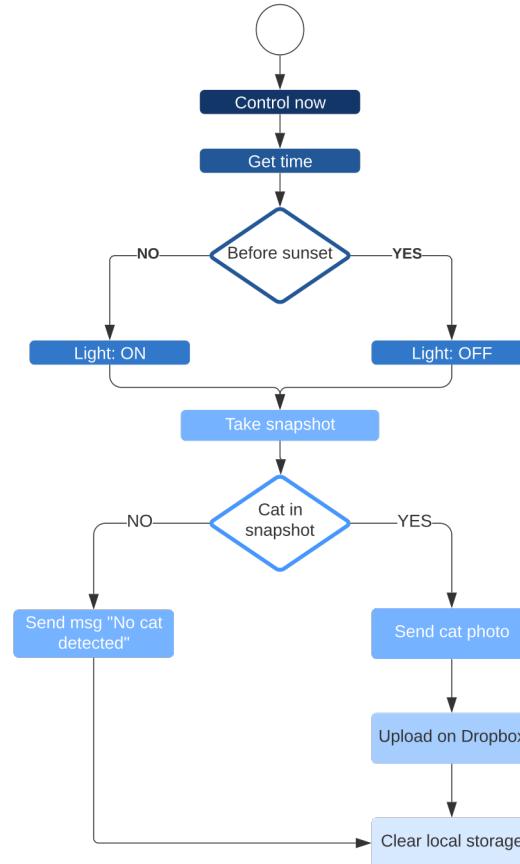
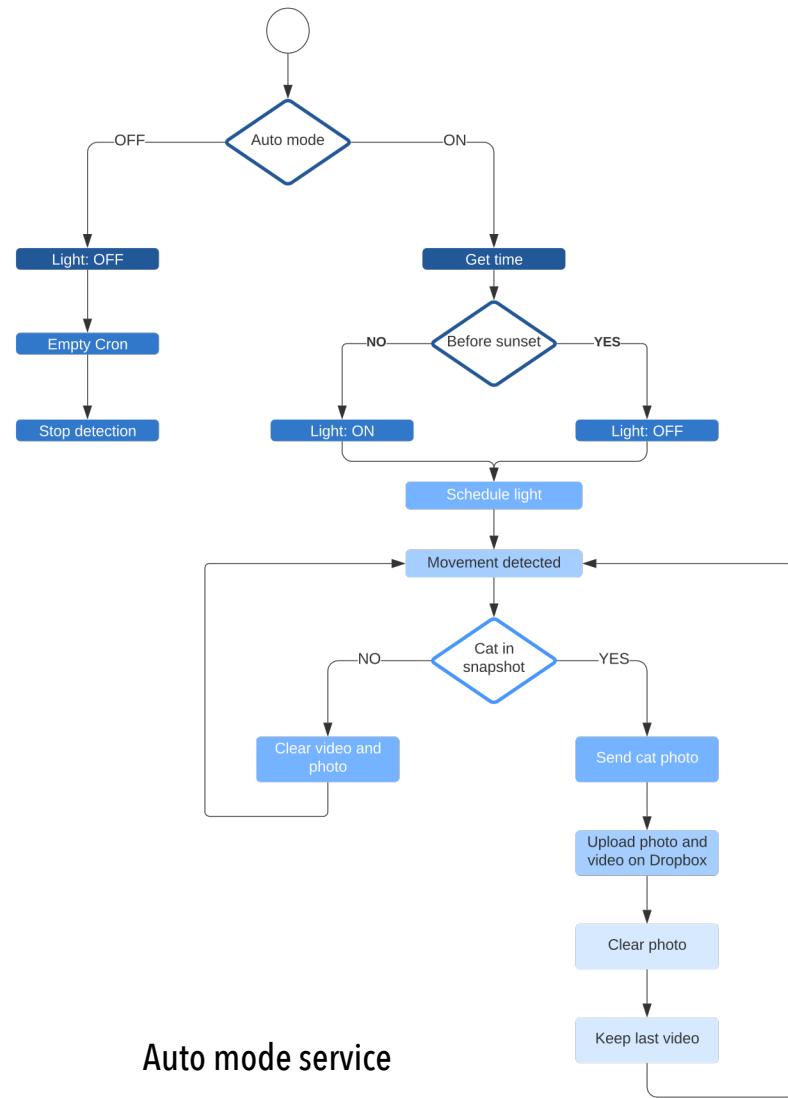
```
#snapshot mode
if last_image_path.endswith("snapshot.jpg"):
    if not cat:
        bot.send_message(chat_id=CHAT_ID, text="Cat not detected", reply_markup=get_Keyboard())
    else:
        bot.send_chat_action(chat_id=CHAT_ID, action=ChatAction.UPLOAD_PHOTO)
        bot.send_photo(chat_id=CHAT_ID, photo=last_image, caption="Cat detected!", reply_markup=get_Keyboard())
        last_image.seek(0)
        upload_to_dropbox(last_image,os.path.basename(last_image_path),type='/snapshots/')
        os.remove(last_image_path)
```

- Viene quindi eseguito automaticamente lo script *on_picture_save*, che verifica se è presente o meno un gatto nella foto e risponde all'utente. Salvataggio su cloud dello snapshot e gestione lampadina presenti.

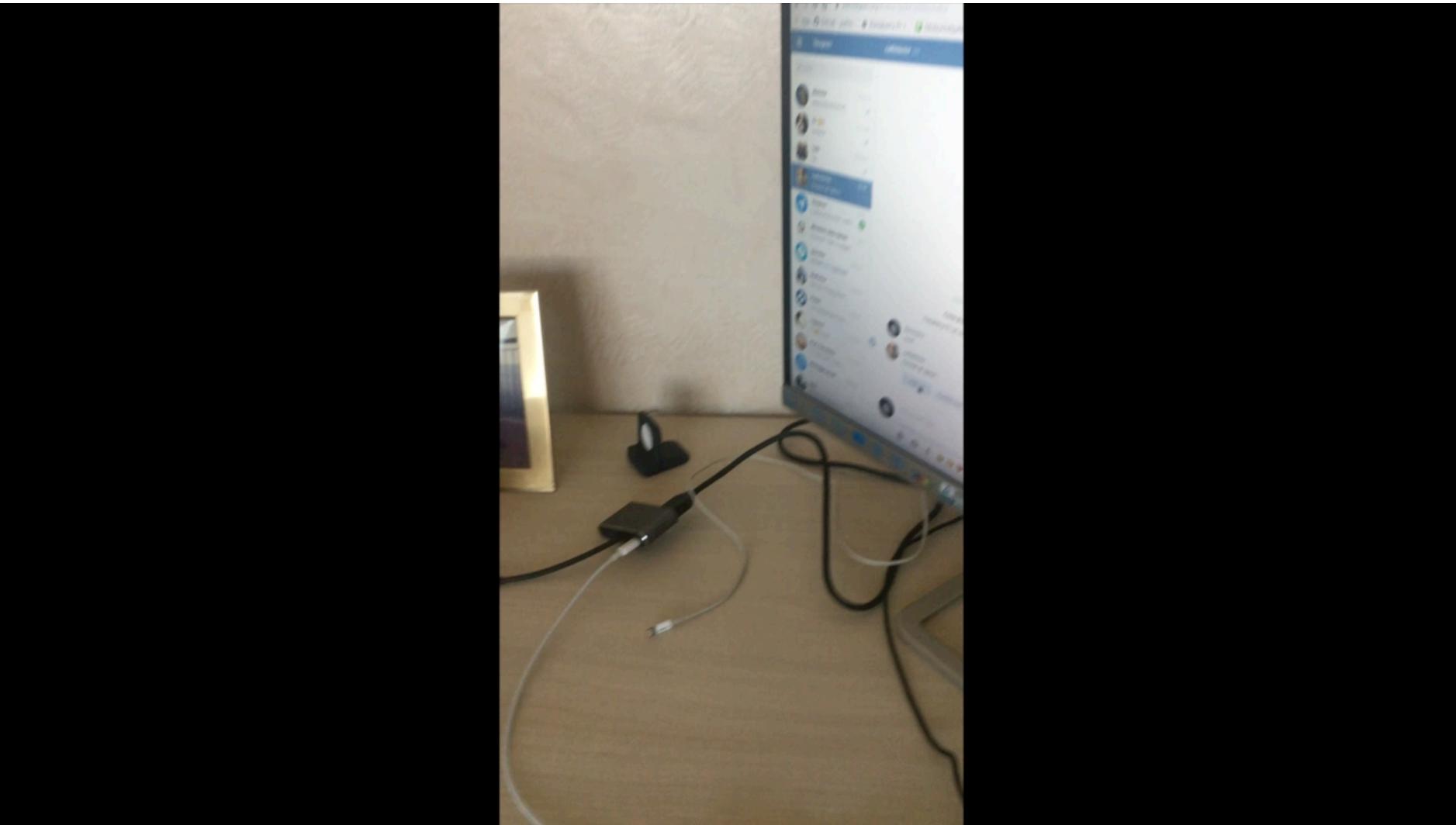
Altre opzioni (3)

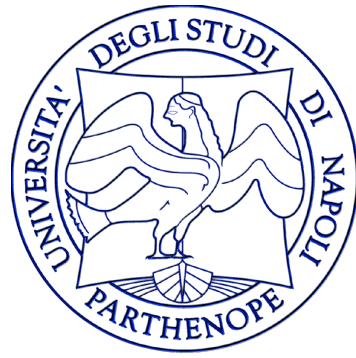
- **Live stream:** settando il parametro *stream_localhost* a off e impostando una porta tramite il parametro *stream_port* all'interno del file *motion.conf* è possibile utilizzare Motion per visualizzare il live stream della Raspberry Camera su rete locale. In caso di utilizzo esterno è possibile ricorrere ad una VPN.

Flow diagrams



Demo





Dipartimento di Scienze e Tecnologie
Corso di laurea in Informatica Applicata

Grazie per l'attenzione!

Studente: Domenico Scognamiglio
Matricola: 0120/000169