

<<<<

AIR CURE

>>>>

TEAM JUNIOR GHOSTS

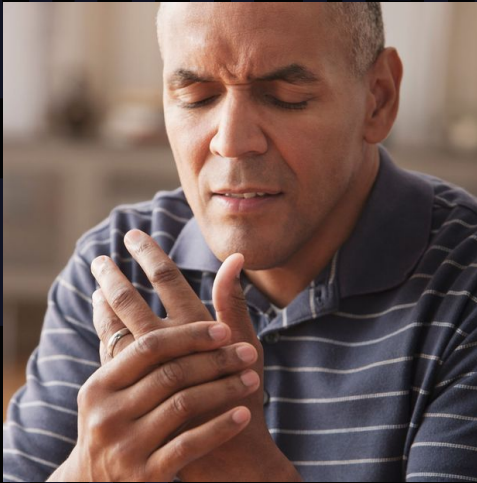
Extremely Low Cost Gamified Therapy

Mohammed // Fazith // Raja

PROBLEM

Physiotherapy Wellness Centers are

- ❖ Not Affordable enough
- ❖ Not Engaging enough
- ❖ Not Well Known enough



1 in **4** Adults will suffer from Arthritis in lifetime

That's 550 million people in SG!

PROBLEM

Physiotherapy Wellness Centers are

- ❖ **Not Affordable** enough
- ❖ **Not Engaging** enough
- ❖ **Not Well Known** enough



Medication and Therapy cost **SGD\$130/Month**

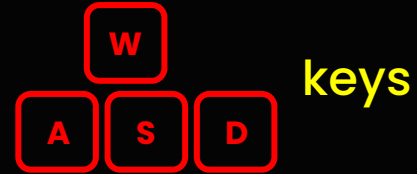
Physiotherapy gloves cost over 300 SGD

Solution

A **Gamified Gesture Based Approach** to actively promote physiotherapy for Arthritis and Carpal Tunnel

Low Cost at **only SGD\$1***

Ability to play **ANY** online game that uses



<<<<

UNLIMITED GAMES || UNLIMITED FUN

(* Cost is derived from weight of 3D print used in glove)

TRAINING - BEFORE IMPROVEMENTS

```
Import TensorFlow and other libraries

In [2]: import matplotlib.pyplot as plt
import numpy as np
import os
import PIL
import tensorflow as tf
import pathlib
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential

Check how many images in images folder

In [3]: image_data_dir = './images'
image_data_dir = pathlib.Path(image_data_dir)
image_count = len(list(image_data_dir.glob('*/*.png')))
image_count_palm = len(list(image_data_dir.glob('palm/*.png')))
image_count_one = len(list(image_data_dir.glob('one/*.png')))
image_count_two = len(list(image_data_dir.glob('two/*.png')))
image_count_three = len(list(image_data_dir.glob('three/*.png')))
image_count_four = len(list(image_data_dir.glob('four/*.png')))
image_count_five = len(list(image_data_dir.glob('five/*.png')))
image_count_c = len(list(image_data_dir.glob('c/*.png')))
image_count_l = len(list(image_data_dir.glob('l/*.png')))
image_count_ok = len(list(image_data_dir.glob('ok/*.png')))
image_count_thumb = len(list(image_data_dir.glob('thumb/*.png')))
print('Total = ' + str(image_count))
print('palm = ' + str(image_count_palm))
print('one = ' + str(image_count_one))
print('two = ' + str(image_count_two))
print('three = ' + str(image_count_three))
print('four = ' + str(image_count_four))
print('five = ' + str(image_count_five))
print('c = ' + str(image_count_c))
print('l = ' + str(image_count_l))
print('ok = ' + str(image_count_ok))
print('thumb = ' + str(image_count_thumb))

Total = 3600
palm = 360
one = 360
two = 360
three = 360
four = 360
five = 360
c = 360
l = 360
ok = 360
thumb = 360
```

CNN Sequential Model
3600 Total Pics

```
In [12]: AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)

The RGB channel values are in the [0, 255] range. This is not ideal for a neural network, in general we should seek to make
your input values small. Here, we will standardize values to be in the [0, 1] range by using a Rescaling layer.

In [13]: normalization_layer = layers.experimental.preprocessing.Rescaling(1./255)

In [14]: normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
# Notice the pixels values are now in `[0,1]`.
print(np.min(first_image), np.max(first_image))

0.0005286801 1.0
```

Used cache to speed up image
loading



Overfitting
Problem: Not enough Dataset

TRAINING - AFTER IMPROVEMENTS

Data augmentation

```
In [22]: data_augmentation = keras.Sequential([
    layers.experimental.preprocessing.RandomFlip("horizontal",
        input_shape=(img_height,
            img_width,
            3)),
    layers.experimental.preprocessing.RandomRotation(0.1),
    layers.experimental.preprocessing.RandomZoom(0.1),
])
```

Visualize few augmented examples look like by applying data augmentation to the same image several times.

```
In [23]: plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
    for i in range(9):
        augmented_images = data_augmentation(images)
        ax = plt.subplot(3, 3, 1 + i)
        plt.imshow(augmented_images[0].numpy().astype("uint8"))
        plt.axis("off")
```



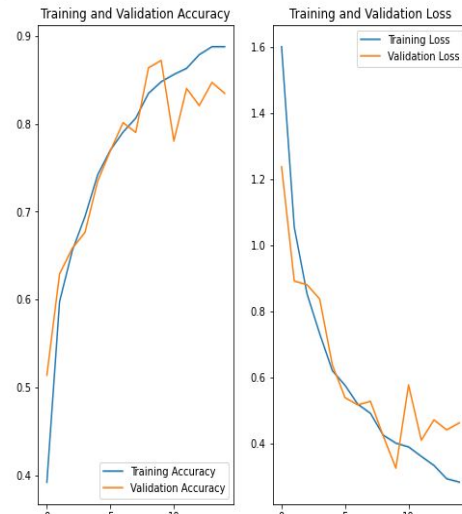
Data Augmentation to
generate additional
Dataset

Dropout and retrain

Apply Dropout to a layer if randomly drops out (by setting the activation to zero) a number of output units from the layer during the training process. Dropout takes a fractional number as its input value, in the form such as 0.1, 0.2, 0.4, etc. This means dropping out 10%, 20% or 40% of the output units randomly from the applied layer. Create a new neural network using layers.Dropout, then train it using augmented images.

```
In [24]: model = Sequential([
    data_augmentation,
    layers.experimental.preprocessing.Rescaling(1./255),
    layers.Conv2D(16, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(32, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(64, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Dropout(0.2),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(num_classes)
])
```

Implemented Dropout
regularization for nodes



Nice Fitted Model

Train accuracy: 89.6%
Test accuracy: 83.47%

The image shows a code editor with a file explorer on the left. The code is a Python script for gesture prediction using a neural network. It includes imports for cv2, numpy, and time. The main function is a loop that processes video frames from a webcam, detecting faces and predicting gestures based on the position of the hand. The code is as follows:

```

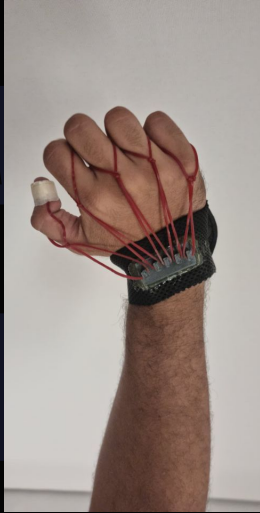
1 # Drowsy.py
2 # -*- coding: utf-8 -*-
3 # Readme and
4 # Requirements list
5 #
6 #
7 #
8 #
9 #
10 #
11 #
12 #
13 #
14 #
15 #
16 #
17 #
18 #
19 #
20 #
21 #
22 #
23 #
24 #
25 #
26 #
27 #
28 #
29 #
30 #
31 #
32 #
33 #
34 #
35 #
36 #
37 #
38 #
39 #
40 #
41 #
42 #
43 #
44 #
45 #
46 #
47 #
48 #
49 #
50 #
51 #
52 #
53 #
54 #
55 #
56 #
57 #
58 #
59 #
60 #
61 #
62 #
63 #
64 #
65 #
66 #
67 #
68 #
69 #
70 #
71 #
72 #
73 #
74 #
75 #
76 #
77 #
78 #
79 #
80 #
81 #
82 #
83 #
84 #
85 #
86 #
87 #
88 #
89 #
90 #
91 #
92 #
93 #
94 #
95 #
96 #
97 #
98 #
99 #
100 #
101 #
102 #
103 #
104 #
105 #
106 #
107 #
108 #
109 #
110 #
111 #
112 #
113 #
114 #
115 #
116 #
117 #
118 #
119 #
120 #
121 #
122 #
123 #
124 #
125 #
126 #
127 #
128 #
129 #
130 #
131 #
132 #
133 #
134 #
135 #
136 #
137 #
138 #
139 #
140 #
141 #
142 #
143 #
144 #
145 #
146 #
147 #
148 #
149 #
150 #
151 #
152 #
153 #
154 #
155 #
156 #
157 #
158 #
159 #
160 #
161 #
162 #
163 #
164 #
165 #
166 #
167 #
168 #
169 #
170 #
171 #
172 #
173 #
174 #
175 #
176 #
177 #
178 #
179 #
180 #
181 #
182 #
183 #
184 #
185 #
186 #
187 #
188 #
189 #
190 #
191 #
192 #
193 #
194 #
195 #
196 #
197 #
198 #
199 #
200 #
201 #
202 #
203 #
204 #
205 #
206 #
207 #
208 #
209 #
210 #
211 #
212 #
213 #
214 #
215 #
216 #
217 #
218 #
219 #
220 #
221 #
222 #
223 #
224 #
225 #
226 #
227 #
228 #
229 #
230 #
231 #
232 #
233 #
234 #
235 #
236 #
237 #
238 #
239 #
240 #
241 #
242 #
243 #
244 #
245 #
246 #
247 #
248 #
249 #
250 #
251 #
252 #
253 #
254 #
255 #
256 #
257 #
258 #
259 #
260 #
261 #
262 #
263 #
264 #
265 #
266 #
267 #
268 #
269 #
270 #
271 #
272 #
273 #
274 #
275 #
276 #
277 #
278 #
279 #
280 #
281 #
282 #
283 #
284 #
285 #
286 #
287 #
288 #
289 #
290 #
291 #
292 #
293 #
294 #
295 #
296 #
297 #
298 #
299 #
300 #
301 #
302 #
303 #
304 #
305 #
306 #
307 #
308 #
309 #
310 #
311 #
312 #
313 #
314 #
315 #
316 #
317 #
318 #
319 #
320 #
321 #
322 #
323 #
324 #
325 #
326 #
327 #
328 #
329 #
330 #
331 #
332 #
333 #
334 #
335 #
336 #
337 #
338 #
339 #
340 #
341 #
342 #
343 #
344 #
345 #
346 #
347 #
348 #
349 #
350 #
351 #
352 #
353 #
354 #
355 #
356 #
357 #
358 #
359 #
360 #
361 #
362 #
363 #
364 #
365 #
366 #
367 #
368 #
369 #
370 #
371 #
372 #
373 #
374 #
375 #
376 #
377 #
378 #
379 #
380 #
381 #
382 #
383 #
384 #
385 #
386 #
387 #
388 #
389 #
390 #
391 #
392 #
393 #
394 #
395 #
396 #
397 #
398 #
399 #
400 #
401 #
402 #
403 #
404 #
405 #
406 #
407 #
408 #
409 #
410 #
411 #
412 #
413 #
414 #
415 #
416 #
417 #
418 #
419 #
420 #
421 #
422 #
423 #
424 #
425 #
426 #
427 #
428 #
429 #
430 #
431 #
432 #
433 #
434 #
435 #
436 #
437 #
438 #
439 #
440 #
441 #
442 #
443 #
444 #
445 #
446 #
447 #
448 #
449 #
450 #
451 #
452 #
453 #
454 #
455 #
456 #
457 #
458 #
459 #
460 #
461 #
462 #
463 #
464 #
465 #
466 #
467 #
468 #
469 #
470 #
471 #
472 #
473 #
474 #
475 #
476 #
477 #
478 #
479 #
480 #
481 #
482 #
483 #
484 #
485 #
486 #
487 #
488 #
489 #
490 #
491 #
492 #
493 #
494 #
495 #
496 #
497 #
498 #
499 #
500 #
501 #
502 #
503 #
504 #
505 #
506 #
507 #
508 #
509 #
510 #
511 #
512 #
513 #
514 #
515 #
516 #
517 #
518 #
519 #
520 #
521 #
522 #
523 #
524 #
525 #
526 #
527 #
528 #
529 #
530 #
531 #
532 #
533 #
534 #
535 #
536 #
537 #
538 #
539 #
540 #
541 #
542 #
543 #
544 #
545 #
546 #
547 #
548 #
549 #
550 #
551 #
552 #
553 #
554 #
555 #
556 #
557 #
558 #
559 #
560 #
561 #
562 #
563 #
564 #
565 #
566 #
567 #
568 #
569 #
570 #
571 #
572 #
573 #
574 #
575 #
576 #
577 #
578 #
579 #
580 #
581 #
582 #
583 #
584 #
585 #
586 #
587 #
588 #
589 #
590 #
591 #
592 #
593 #
594 #
595 #
596 #
597 #
598 #
599 #
600 #
601 #
602 #
603 #
604 #
605 #
606 #
607 #
608 #
609 #
610 #
611 #
612 #
613 #
614 #
615 #
616 #
617 #
618 #
619 #
620 #
621 #
622 #
623 #
624 #
625 #
626 #
627 #
628 #
629 #
630 #
631 #
632 #
633 #
634 #
635 #
636 #
637 #
638 #
639 #
640 #
641 #
642 #
643 #
644 #
645 #
646 #
647 #
648 #
649 #
650 #
651 #
652 #
653 #
654 #
655 #
656 #
657 #
658 #
659 #
660 #
661 #
662 #
663 #
664 #
665 #
666 #
667 #
668 #
669 #
670 #
671 #
672 #
673 #
674 #
675 #
676 #
677 #
678 #
679 #
680 #
681 #
682 #
683 #
684 #
685 #
686 #
687 #
688 #
689 #
690 #
691 #
692 #
693 #
694 #
695 #
696 #
697 #
698 #
699 #
700 #
701 #
702 #
703 #
704 #
705 #
706 #
707 #
708 #
709 #
710 #
711 #
712 #
713 #
714 #
715 #
716 #
717 #
718 #
719 #
720 #
721 #
722 #
723 #
724 #
725 #
726 #
727 #
728 #
729 #
730 #
731 #
732 #
733 #
734 #
735 #
736 #
737 #
738 #
739 #
740 #
741 #
742 #
743 #
744 #
745 #
746 #
747 #
748 #
749 #
750 #
751 #
752 #
753 #
754 #
755 #
756 #
757 #
758 #
759 #
760 #
761 #
762 #
763 #
764 #
765 #
766 #
767 #
768 #
769 #
770 #
771 #
772 #
773 #
774 #
775 #
776 #
777 #
778 #
779 #
780 #
781 #
782 #
783 #
784 #
785 #
786 #
787 #
788 #
789 #
790 #
791 #
792 #
793 #
794 #
795 #
796 #
797 #
798 #
799 #
800 #
801 #
802 #
803 #
804 #
805 #
806 #
807 #
808 #
809 #
810 #
811 #
812 #
813 #
814 #
815 #
816 #
817 #
```

Mapped prediction of gestures to Keyboard W A S D using PyKey and Press libraries

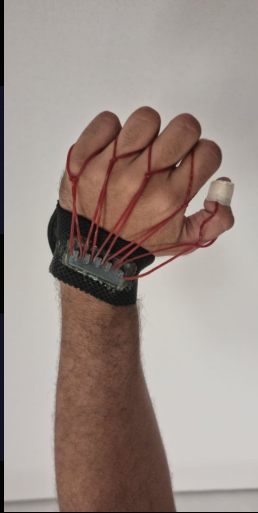


Used **Flask web development** and **OpenCV** to demonstrate running desired game (*in this case: Super Mario*)

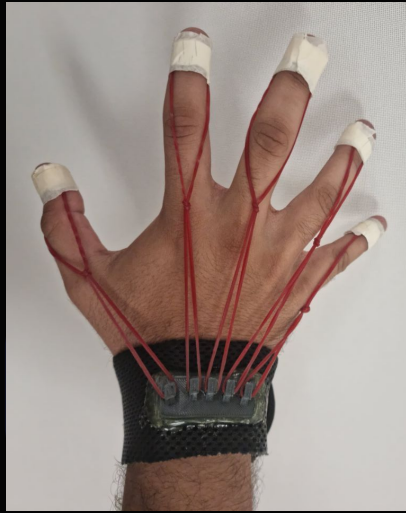
OUR PROJECT - 3D PRINTED PHYSIOTHERAPY GLOVE



"A"
(Go Left)



"D"
(Go Right)



"W"
(Go Straight/Jump)



**Another orientation
to train upper hand
muscles**

FUTURE DEVELOPMENT - BETTER AI DETECTION MODEL

Due to Time Constraints, We weren't able to completely fine tune AI Model

```
In [15]: model.compile(optimizer='adam',  
                        loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),  
                        metrics=['accuracy'])
```

(RIGHT NOW)



Otsu's method



All

Images

Videos

News

Shopping

More

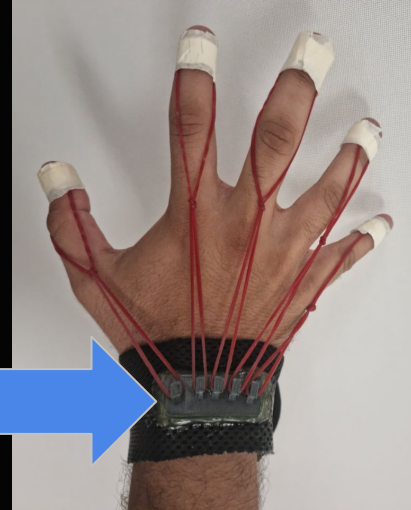
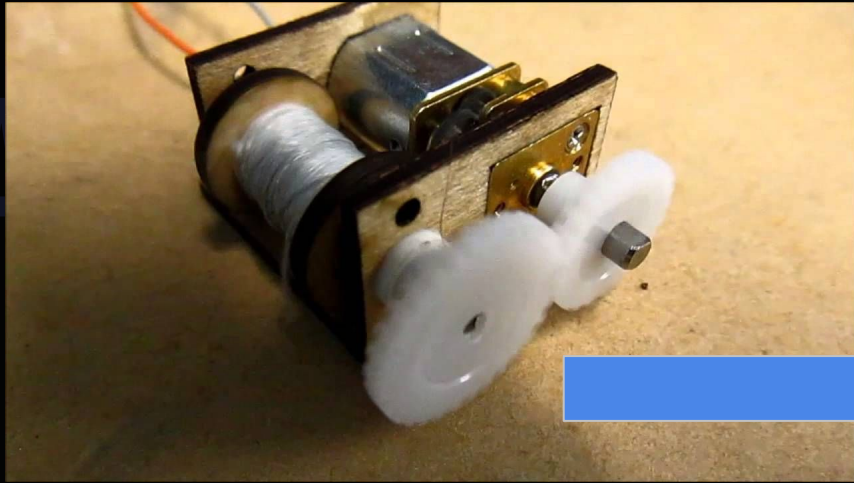
About 173,000 results (0.65 seconds)

Otsu's method[1] is a **variance-based technique to find the threshold value where the weighted variance between the foreground and background pixels is the least**. The key idea here is to iterate through all the possible values of threshold and measure the spread of background and foreground pixels.

13 Mar 2020

(IN FUTURE)

FUTURE DEVELOPMENTS - HARDWARE



3D Printed Winches on each end of the bands for personalised adjustable tension

FUTURE DEVELOPMENTS - ASSISTIVE TECH FOR STUDENTS



- We can also propel our product towards learning for students with disabilities
- Allows them to learn/ do assignments through assistive technology provided by AIRCure
- Eliminates Teaching Assistant writing/doing things the student asks them to write

OUR PROJECT - BUSINESS/COMMERCIAL VIABILITY



- 1) Partner Local Physiotherapy Clinics**
- 2) 1\$ affordable therapy gloves**
- 3) Access to MILLIONS of online games**

SUPREMELY VERSATILE TEAM

SP Singapore
Polytechnic



Mohammed HK
Diploma in Computer Engineering



Fazith Ismail
Diploma in Computer Engineering



Raja
Diploma in Computer Engineering

Engaging therapy for thousands of people in Singapore