2024

Tactigon
Skin Project

Ram, Jem, Fum, Ted, El W

THE TACTIGON™
Skin

Intuitive: Recognizes natural hand movements.

Connected: Integrates with IoT, VR/AR, and more**.**

Versatile: Applications in robotics, gaming, and remote control.

“The wearable for gesture control"

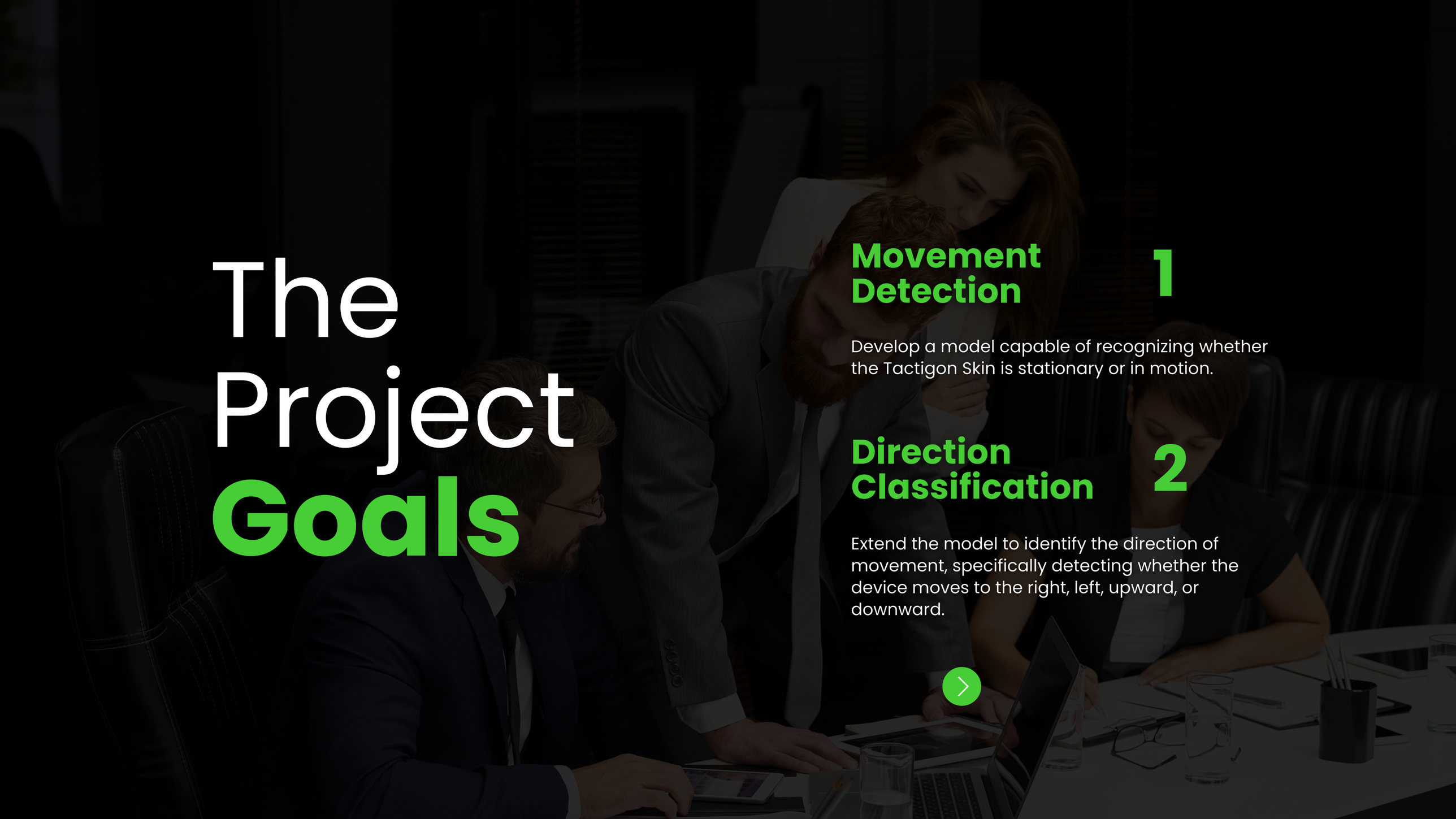
Technologies and Requirements

Main Requirements:

1. Reliable **movement** detection.
2. Real-time data **processing**.
3. Easy **integration** into various applications.

Technologies Used:

1. Tactigon Skin **sensors**: accelerometer, gyroscope, and touch sensor.
2. Tactigon Gear **library** for data acquisition.
3. Python framework: **TensorFlow** for neural network models.
4. Standard neural networks for motion **prediction**.



The Project **Goals**

Movement Detection 1

Develop a model capable of recognizing whether the Tactigon Skin is stationary or in motion.

Direction Classification 2

Extend the model to identify the direction of movement, specifically detecting whether the device moves to the right, left, upward, or downward.

2024

Our First Strategy

Data Creation

Data gathered using the Tactigon library.

Movements labeled as stationary or in motion.

Dataset structured for training the neural network.

Binary Classification Model

Distinguish between stationary and in motion states.

Input: 6 features (accelerometer and gyroscope data).

Model: Dense Neural Network (Dense NN).

Output: Binary category (stationary or in motion).

Data Streaming Issues

The Tactigon Skin streams data in a continuous loop.

Issue: Grouping 5 instances (100 ms) for the model while maintaining real-time synchronization.

Consequence: The model could not be properly tested in real-time, only on static datasets.

Second Strategy

New Strategy

Grouped data into blocks of 5 instances (100 ms total).

Values combined into tuples to represent complete movements instead of single moments.

4 Labels

Movements: **up, down, left, right.**

Any other value: **not moving.**

Results

Using a Machine Learning model

Successfully distinguished between moving and not moving.

Issue: Classification of movement directions was not accurate.

The model detected direction changes, but there were **errors** in label assignment.

Possible Strategies

1) Input Sampling: Sampling when the movement is detected

2) Reshaping and labeling actions

3) Undestanding multiprocessing

4) Making triggers w/ Tactigon Skin

Our Last **Strategy**

We created a way to stop and start record.
We made a simple dataset.
We trained a new neural network.
We fixed many bugs and optmized the visualisation.
Now... let's see how it works!

Completed

Our Team

Andrea Fumagalli

Davide Jemtchougov

Adish Ramrutton

Andrea Tedesco

Ilies El Wiyadi

2024