July 10, 2019

DETECTING ATTRIBUTES OF USER INPUTS UTILIZING MOTION SENSOR DATA AND MACHINE LEARNING

Philip Quinn

Michael Xuelin Huang

Shumin Zhai

DETECTING ATTRIBUTES OF USER INPUTS UTILIZING MOTION SENSOR DATA AND MACHINE LEARNING

ABSTRACT

A computing device is described that uses motion data from motion sensors to detect user inputs, such as out-of-screen user inputs for mobile devices. In other words, the computing device detects user touch inputs at locations of the device that do not include a touch screen, such as anywhere on the surface of the housing or case of the device. The techniques described enable a computing device to utilize a standard, existing motion sensor (e.g., an inertial measurement unit, (IMU), accelerometer, gyroscope, etc.) to detect the user input and determine attributes of the user input. Motion data generated by the motion sensor (also referred to as a movement sensor) is processed by an artificial neural network to infer characteristics or attributes of the user input, including: a location on the housing where the input was detected, a surface of the housing where the input was detected (e.g., front, back, and edges, such as top, bottom and sides); a type of user input (e.g., finger, stylus, fingernail, finger pad, etc.). In other words, the computing device applies a machine-learned model to the sensor data to classify or label the various attributes, characteristics, or qualities of the input. In this way, the computing device utilizes machine learning and motion data to classify attributes of the user input or gesture utilizing motion sensors without the need for additional hardware, such as touch-sensitive devices and sensors.

DESCRIPTION

Techniques are described that enable a computing device to detect gestures performed on a surface of a computing device that is not touch-sensitive, such as the back of the computing device, using data from one or more motion sensors. The computing device may differentiate between gestures on different surfaces (e.g., front versus back) based on motion data generated

by one or more motion sensors (e.g., an accelerometer and/or a gyroscope). In some examples, the computing device differentiates between gestures at different locations on a given surface at a relatively high-level of granularity based on motion data generated by one or more motion sensors. For example, the computing device may detect gestures in 25, 50, 100, or more locations on a surface of the device. In yet another example, the computing device differentiates between touch input devices based on motion data generated by one or more motion sensors. For example, the computing device may differentiate between gestures performed with a pad of a user's finger and gestures performed with a fingernail. In this way, the computing device detects gestures, even at surfaces of the computing device that do not include a touch-sensitive device, based on the motion of the computing device. Utilizing the motion data may enable the computing device to detect gestures at surfaces of the computing device that do not include a touch-sensitive device. In some scenarios, the computing device may utilize the motion data to detect gestures at surfaces that do include a touch-sensitive surface without activating the touch-sensors, which may reduce the amount of power consumed by the computing device.

In the example of FIG. 1, computing device 2 is configured to detect user inputs on the front, back, and/or sides of computing device 2 based at least in part on data generated by one or more motion sensors. Examples of computing device 2 include, but are not limited to, smartphones, tablets, watches, fitness trackers (e.g., wrist-worn, ankle-worn, waist-worn, or other devices worn by users that track various movement or other athletic activity), counter top computing devices, laptops, or any other computing device. Computing device 2 includes a touch-sensitive device (e.g., a touchscreen) on the front, one or more processors, and one or more motion sensors. Examples of processors include, but are not limited to, digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry.

Examples of motion sensors include an accelerometer, gyroscope, inertial measurement unit (IMU), magnetometer, among others.

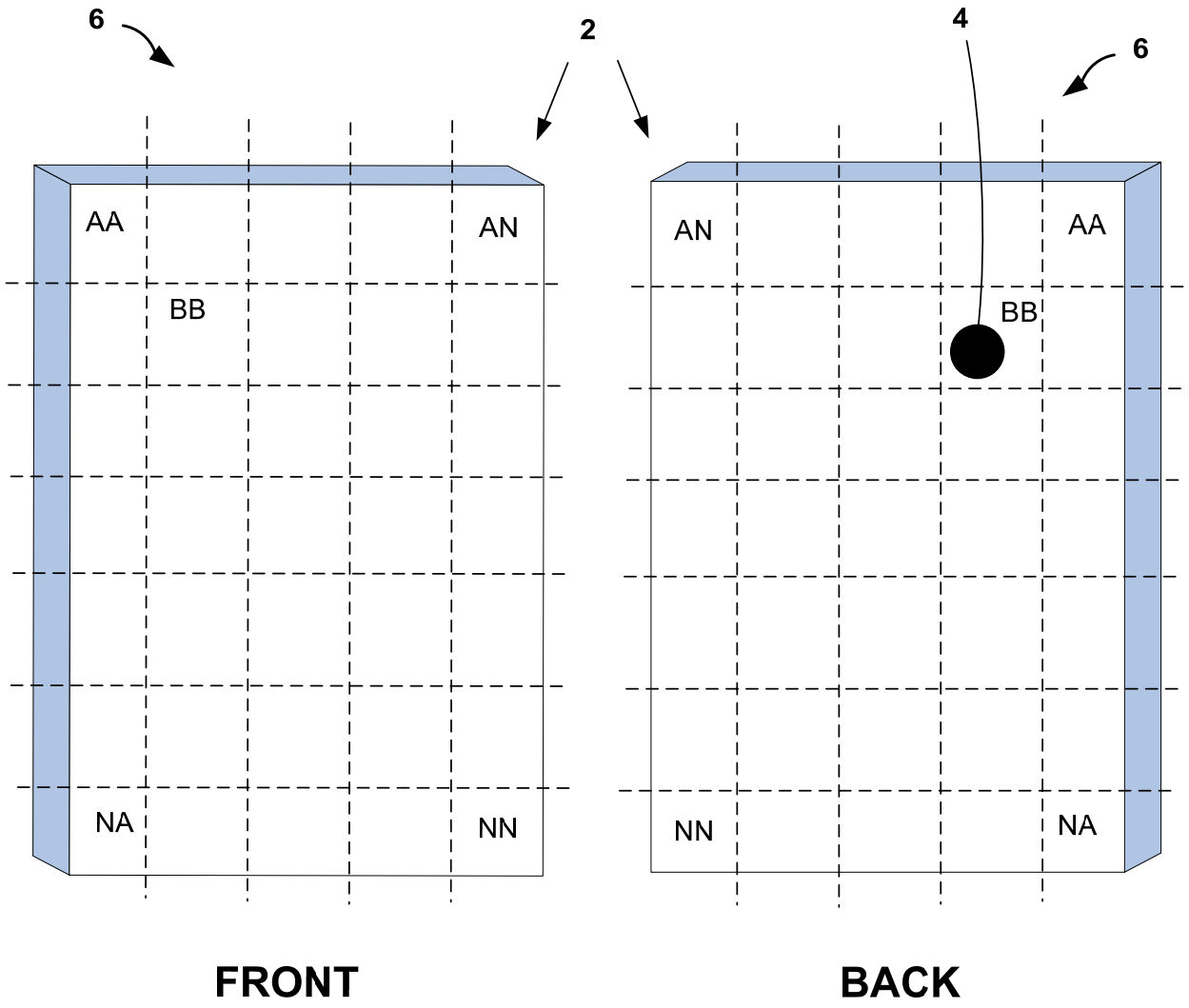

FIG. 1

In some examples, the front of computing device 2 may include a touch-sensitive device while the back of computing device 2 does not include a touch-sensitive device. In other examples, the front and back of computing device 2 both include touch-sensitive devices. Examples of touch-sensitive devices include touchscreens and touchpads, among others.

A user of computing device 2 may perform a gesture on the front, back, and/or sides of computing device 2. In other words, the user may perform a gesture or user input anywhere

along the housing or case of the computing device 2. As one example, the user may perform a tap, double-tap, swipe, pinch, or any other type of gesture at the back of computing device 2 (e.g., using his/her finger, a stylus, or other input mechanism that makes physical contact with computing device 2. As illustrated in FIG. 1, a user of computing device 2 may tap the back surface of the housing of computing device 2 at location 4.

Computing device 2 includes one or more motion sensors configured to detect motion of computing device 2 as the input device contacts the back of computing device 2 at location 4. In one example, computing device 2 may include an IMU, which may include a 3-axis accelerometer and a 3-axis gyroscope. In such examples, the IMU outputs sensor data with six values (e.g., an acceleration value for each of the three axes and a gyroscope value for each of the three axes) each time IMU samples the motion of computing device 2. Computing device 2 may store the sensor data as one-dimensional vector (or as any other type of data structure) with six sensor values or elements for each time the IMU samples the motion. The IMU may sample the motion every 1 millisecond, every 2 milliseconds, every 5 milliseconds, and so on. In some examples, the vector includes data for multiple samples (e.g., 10, 20, 50, 100 samples). In this way, the vector may include six sensor values for each of the plurality of samples.

In accordance with techniques of this disclosure, computing device 2 applies the sensor data to a machine-learned model to determine the attributes or characteristics of the touch input. The model may be trained via supervised or unsupervised learning. In some examples, the model includes a neural network, such as a convolutional neural network (CNN). Additional examples of machine learning algorithms include clustering algorithms, decision-tree algorithms, regression algorithms, as only a few examples. In one example, computing device 2 may apply the vector of sensor data to the CNN. In other words, the input to the CNN includes, in some examples, a one-dimensional vector (e.g., with six values for each time the IMU is sampled).

Computing device 2 may apply a multi-layer CNN to the input vector. For example, the CNN may include 2, 3, 4, or more layers. The layers may be partially or fully connected. For each convolution layer, computing device 2 applies a filter to the input for that layer. As one example, computing device 2 may multiply the one-dimensional input vector (also referred to as a matrix) by a convolution filter and output a convoluted vector (or matrix) for the first convolution layer. Computing device 2 may input the convoluted vector output by the first layer as an input to the second layer, apply another filter, and output a new convoluted vector, and so on for each convolution layer of the CNN.

Computing device 2 may determine a plurality of attributes of the user input based on the CNN. In other words, in one example, the CNN receives the one-dimensional vector (e.g., with six values for each sample of the IMU) as its input and outputs a plurality of values representative of the attributes of the user input. In some examples, the attributes include a coarse location at which the user input was received, a surface at which the user input was received (e.g., front, back, top, or bottom), a type of input device, and a refined or specific location at which the user input was received.

In some examples, because different attributes of the user input may affect the sensor data simultaneously, computing device 2 may apply the multi-layer CNN to jointly learn the mappings from sensor data onto these interrelated attributes in a multi-task learning paradigm. In one example, computing device 6 determines the different attributes of the user input by utilizing one or more shared convolutional layers, which may enable computing device 2 to detect or extract common patterns that are indicative across user input attributes. Following the shared layers, in some examples, the CNN includes individual layers for each attribute of the user input to extract the attribute-dependent patterns.

In some examples, one or more outputs of the CNN include classifications of the attributes or characteristics of the input. For example, computing device 2 may classify the

coarse location at which the user input occurred as a grid element AA-NN of grid 6. In the example of FIG. 1, a first output of the CNN classifies the coarse location of the user input as grid location BB. As another example, a second output of the CNN classifies the surface at which the user input was received as a back surface of computing device 2. The third output of the CNN may classify a type of the input device. Examples types of input devices include a finger, a stylus, or a pen, among others. In some example, the CNN classifies a finger into sub-classes, such as a fingernail or a finger-pad. That is, the third output of the CNN may classify the type of input device as a fingernail or finger-pad.

One or more outputs of the CNN include, in some instances, a regression output. For example, a fourth output of the CNN may include a refined or specific location at the user input was received. That is, while computing device 2 may utilize a CNN to classify a location of the user input as a particular grid element of grid 6, computing device 2 may additionally or alternatively determine a specific location at which the user input was received.

Computing device 6 may perform one or more actions based on the classifications of the characteristics of the user input. For example, computing device 6 may select an icon (e.g., a folder icon, application icon, etc.) displayed on the front of computing device 2 at grid location BB of grid 6 in response to determining the user input was received at grid location BB on the back of the device.

In some instances, computing device 6 may perform different actions based on the type of the user input. For instance, computing device 6 may open a folder for a folder icon displayed at grid location BB if the type of user input was a finger-pad and may display additional options associated with the folder icon, such as a context menu (e.g., a menu of options such as delete, rename, etc.) if the type of user input was a fingernail. In this way, computing device 6 may perform different actions based on the type of user input in a manner similar to performing different actions for a right-click versus left-click, a hard-press vs a light-press, or a long-press vs

a short press. Similarly, computing device 6 may perform different actions depending on the surface at which the input was located.

By applying motion data to a machine-learned model, such as a CNN model, computing device 2 may determine one or more attributes of a touch input using existing motion sensors. Utilizing motion sensors to detect touch inputs may enable the computing device to determine attributes of touch inputs without utilizing a touch-sensitive device, such as a touchscreen or touchpad. Detecting user inputs and determining the attributes of the touch input without utilizing a touch-sensitive device may the cost of the computing device and/or reduce the power consumed by the device, for example, by detecting user inputs without activating a touch-sensitive device. Further, detecting user inputs without utilizing a touch-sensitive device may enable the computing device to detect user inputs at locations of the computing device that do not include a touch-sensitive device, which may open the door to new functionality of the computing device.

REFERENCES

1. U.S. Patent No. 9,235,278 entitled “Machine-learning based tap detection” to Cheng, et al.

2. U.S. Patent No. 9,134,818 entitled “Isolating mobile device electrode” to Hughes, et al.

3. U.S. Patent Publication No. 2017/0,076,195 entitled “Distributed neural networks for scalable real-time analytics” by Yang et al.