

device employed herein further includes a particular coding pattern so that: 1) adjacent electrodes never share the same sensor circuit; and 2) the electrodes sharing the same sensor circuit are always separated from one another by the dispersal distance, i.e., roughly one third of the number of electrodes.

[0013] A touch sensitive device incorporating the teachings herein is illustrated in FIG. 1. The capacitive touch sensitive device 100 is a one-dimensional circular array, although other arrangements, e.g., linear arrays, etc., could also be used. The circular array includes 22 electrodes, numbered 0-21. The circular array includes only 11 sensor circuits. These sensor circuits may take the form of various sensor circuits known to those skilled in the art. One such circuit is disclosed in U.S. Pat. No. 6,323,846, entitled "Method and Apparatus for Integrating Manual Input," which is hereby incorporated by reference. The sensor circuit corresponding to each electrode is designated by a number located at the outer portion of each sensor electrode.

[0014] The touch sensitive device 100 thus shares two electrodes per sensor. However, additional electrodes may be shared with each sensor. Each electrode in FIG. 1 also includes a group designator, either "A" or "B". Each group A electrode shares a sensor with a group "B" electrode. As noted above, the preferred dispersal distance (i.e., the distance between two electrodes sharing a sensor) is a span of approximately one-third the number of sensors, and thus approximately one-third of a characteristic dimension of the device. Thus for the circular device illustrated in FIG. 1, the preferred dispersal distance is approximately one-third the circumference of the circle, thus encompassing approximately one third of the sensors. Any two adjacent electrodes and the two electrodes that share sensor circuits will thus be evenly spaced, a third of the way around the circle. For example, electrode 1 in group A shares sensor 1 with electrode 8 in group B. Electrode 1 is located at approximately the eleven o'clock position, while electrode 8 is located at approximately the seven o'clock position. Similarly, electrode 0 in group A shares sensor 0 with electrode 15 in group B. Electrode 0 is located at the twelve o'clock position, while electrode 15 is located at approximately the four o'clock position.

[0015] The sensor may alternatively be constructed as a one-dimensional linear array. For such a sensor, the dispersal pattern is basically the same as for a circular array: linear arrays can be treated as a circular array that has been broken between two electrodes and uncurled. Again, it is preferred that the dispersal difference between two electrodes sharing a sensor be about one-third the characteristic dimension of the device, which for a linear sensor is the length of the device.

[0016] Obviously, because multiple electrodes share a sensing circuit, the absolute position of an object in contact (proximity) with a single electrode cannot be determined. For absolute position interpolation to work properly in a device constructed according to the principles herein, each electrode must be sufficiently narrow enough that the object being tracked, usually a finger or conductive stylus, overlaps multiple (e.g., two or three) adjacent electrodes. Likewise, to eliminate any ambiguity, the object being tracked must be smaller than the dispersal distance so that it does not overlap both shared electrodes of any sensor circuit.

[0017] While other electrode sharing patterns are possible, some of these can not be used to unambiguously determine the position of a finger. For example, an electrode arrangement with a dispersal distance of half the array size would fail. For a circular array, this would correspond to sharing of electrodes on opposite sides of the circle, 180 degrees from one another. No matter how decoding and interpolation were done, the system could never tell whether the finger or stylus was really at the opposite position halfway around the circle.

[0018] Because each sensor circuit is connected to multiple electrodes, the sensor illustrated herein requires a decoding method that finds the set of electrodes with the largest signals, then decides which of two possible electrode groups would attribute these largest signals to adjacent rather than scattered electrodes. Once this best decoding is known, classic centroid interpolation can commence amongst the adjacent electrodes. For purposes of centroid computation, each sensor's entire signal is attributed to its electrode in the adjacent group, leaving its other electrode from the dispersed group with zero signal and zero contribution to the centroid. Assuming the signal to noise ratio of the sensor circuits is adequate, the sensor described herein offers the same position resolution as a conventional position detector that has a separate sensor circuit for each electrode.

[0019] The example of computer instructions below demonstrates the algorithm used in the present invention to find the position of a finger or stylus that is touching somewhere on the circular array of electrodes. Sensor and electrode mappings are held in look-up-tables (LUTs) to minimize the computation needed for decoding the location of the touching finger. The LUTs map electrode number to sensor number for each group (Sensor_to_A_type_electrode, Sensor_to_B_type_electrode), map the sensor number corresponding to the adjacent electrode (next_X_electrode_sensor, previous_X_electrode_sensor, where X=A or B), and electrode number to sensor number (Electrode_to_Sensor). The use of these LUTs simplifies the calculation of the finger location using the present invention but they are not necessary.

[0020] A brief description of the algorithm implemented by the code is as follows:

[0021] 1. The sensor array is scanned and the signal values corresponding to each sensor are collected.

[0022] 2. The sensor having maximum strength signal is located using code segment findMaxSensor.

[0023] 3. The electrode under which the finger is located is computed using code segment findMaxElectrode.

[0024] 4. The centroid is computed using code segment computeCentroid.

[0025] 5. Steps 1-4 are repeated.

```
#define NUM_SENSORS 11
#define NUM_ELECTRODES 22
// Group A electrode and sensor mappings
Sensor_to_A_type_electrode[NUM_SENSORS] = {0,1,2,3,
4,5,6,7,9,11,13};
next_A_electrode_sensor[NUM_SENSORS] = {1,2,3,4,5,6,7,
```