

duces a vector whose direction can indicate the direction of roll or tilt and whose magnitude can control the rate of roll or tilt about x and y axes.

[0247] Since the weighted and unweighted position averages are only influenced by positions of currently contacting fingers and increases in contact pressure or proximity, the method is insensitive to finger liftoffs. Computation of reference-normalized proximity ratios in step 572 rather than absolute changes in proximity prevents the large palm heel contacts from having undue influence on the weighted average position.

[0248] Since only the current contact positions are used in the average position computations, the roll and tilt vector is independent of lateral motions such as hand translation or rotation as long as the lateral motions do not disturb finger pressure, thus once again achieving integrality. However, hand scaling and differential hand pressure are difficult to use at the same time because flexing the fingers generally causes significant decreases in fingertip contact area and thus interferes with inference of fingertip pressure changes. When this becomes a serious problem, a total hand pressure component can be used as a sixth degree of freedom in place of the hand scaling component. This total pressure component causes cursor velocity along a z-axis in proportion to deviations of the average of the contact proximity ratios from one. Alternative embodiments may include further enhancements such as adapting the reference proximities to slow variations in resting hand pressure and applying a dead zone filter to ignore pressure difference vectors with small magnitudes

[0249] Despite the care taken to measure the polar velocity, translation velocity, and hand pressure components in such a way that the resultant vectors are independent of one another, uneven finger motion during hand scaling, rotation, or translation can still cause minor perturbations in measurements of one degree of freedom while primarily attempting to move in another. Non-linear filtering applied in steps 510 and 512 of FIG. 34 removes the remaining motion leakage between dominant components and nearly stationary components. In steps 510 each component velocity is downscaled by the ratio of its average speed to the maximum of all the component speeds, the dominant component speed:

$$H_{vx}[n] = H_{vx}[n] \times \left(\frac{H_{xyspeed}[n]}{\text{dominantspeed}} \right)^{pds} \quad (74)$$

$$H_{vy}[n] = H_{vy}[n] \times \left(\frac{H_{xyspeed}[n]}{\text{dominantspeed}} \right)^{pds} \quad (75)$$

$$H_{vs}[n] = H_{vs}[n] \times \left(\frac{H_{sspeed}[n]}{\text{dominantspeed}} \right)^{pds} \quad (76)$$

$$H_{vr}[n] = H_{vr}[n] \times \left(\frac{H_{rspeed}[n]}{\text{dominantspeed}} \right)^{pds} \quad (77)$$

where $H_{xyspeed}[n]$, $H_{sspeed}[n]$, and $H_{rspeed}[n]$ are autoregressive averages over time of the translation speed, scaling speed, and rotational speed, where:

$$\text{dominant_speed}[n] = \max(H_{xyspeed}[n], H_{sspeed}[n], H_{rspeed}[n]) \quad (78)$$

where pds controls the strength of the filter. As pds is adjusted towards infinity the dominant component is picked out and all components less than the dominant tend toward zero producing the orthogonal cursor effect well-known in drawing appli-

cations. As pds is adjusted towards zero the filters have no effect. Preferably, pds is set in between so that components significantly slower than the dominant are slowed further, but components close to the dominant in speed are barely affected, preserving the possibility of diagonal motion in multiple degrees of freedom at once. The autoregressive averaging helps to pick out the component or components which are dominant over the long term and suppress the others even while the dominant components are slowing to a stop.

[0250] Step 512 takes a second pass with a related filter known as a dead-zone filter. A dead-zone filter produces zero output velocity for input velocities less than a speed threshold but produces output speeds in proportion to the difference between the input speed and the threshold for input velocities that exceed the threshold. Preferably the speed threshold or width of the dead zone is set to a fraction of the maximum of current component speeds. All velocity components are filtered using this same dead zone width. The final extracted component velocities are forwarded to the chord motion recognizer module 18 which will determine what if any input events should be generated from the motions.

[0251] FIG. 39A shows the details of the finger synchronization detector module 14. The synchronization detection process described below is repeated for each hand independently. Step 600 fetches proximity markers and identifications for the hand's current paths. The identifications will be necessary to ignore palm paths and identify combinations of synchronized fingers, while the proximity markers record the time at which each contact path first exceeds a press proximity threshold and the time at which each contact path drops below a release proximity threshold prior to total liftoff. Setting these proximity thresholds somewhat higher than the minimum proximity considered significant by the segmentation search process 264, produces more precise finger press and release times.

[0252] Step 603 searches for subsets of fingers which touch down at about the same time and for subsets of fingers which lift off at about the same time. This can be done by recording each finger path along with its press time in a temporally ordered list as it crosses the press proximity threshold. Since the primary function of the palms is to support the forearms while the hands are resting, palm activity is ignored by the typing 12 and chord motion recognizers 18 except during differential hand pressure extraction and palm heel presses can be excluded from this list and most other synchronization tests. To check for synchronization between the two most recent finger presses, the press times of the two most recent entries in the list are compared. If the difference between their press times is less than a temporal threshold, the two finger presses are considered synchronized. If not, the most recent finger press is considered asynchronous. Synchronization among three or more fingers up to five is found by comparing press times of the three, four, or five most recent list entries. If the press time of the most recent entry is within a temporal threshold of the nth most recent entry, synchronization among the n most recent finger presses is indicated. To accommodate imprecision in touchdown across the hand, the magnitude of the temporal threshold should increase slightly in proportion to the number of fingers being tested for synchronization. The largest set of recent finger presses found to be synchronized is recorded as the synchronized subset, and the combination of finger identities comprising this subset is stored conveniently as a finger identity bitfield. The term subset is used because the synchronized press subset may not include all fingers