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.

[**0250**] FIG. **39**A 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.

[**0251**] 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 currently touching the surface, as happens when a finger touches down much earlier than the other fingers yet remains touching as they simultaneously touch down. An ordered list of finger release times is similarly maintained and searched separately. Alternative embodiments may require that a finger still be touching the surface to be included in the synchronized press subset.

[**0252**] Decision diamond **602** checks whether a synchronization marker is pending from a previous image scan cycle. If not, decision diamond **604** checks whether the search **603** found a newly synchronized press subset in the current proximity image. If so, step **606** sets the temporal synchronization marker to the oldest press within the new synchronized subset. Additional finger presses may be added to the subset during future scan cycles without affecting the value of this temporal synchronization marker. If there is currently no finger press synchronization, decision diamond **605** determines whether three or more fingers have just been released simultaneously. Simultaneous release of three or more fingers should not occur while typing with a set of fingers but does occur when lifting fingers off the surface from rest. Therefore simultaneous release of three or more fingers reliably indicates that the released fingers are not intended as keypresses and should be deleted from the keypress queue **605**, regardless of whether these same fingers touched down synchronously. Release synchronization of two fingers is not by itself a reliable indicator of typing intent and has no effect on the keypress queue. The keypress queue is described later with FIGS. **42**-**43**B.

[**0253**] Once a press synchronization marker for the hand is pending, further processing checks the number of finger presses which are synchronized and waits for release of the synchronized fingers. If decision diamond **608** finds three or more fingers in the synchronized press subset the user cannot possibly be typing with these fingers. Therefore step **612** immediately deletes the three or more synchronized presses from the keypress queue. This way they cannot cause key symbol transmission to the host, and transmission of key symbols from subsequent asynchronous presses is not blocked waiting for the synchronized fingers to be released.

[**0254**] However, when the synchronization only involves two finger presses **608**, it is difficult to know whether the user intended to tap a finger pair chord or intended to type two adjacent keys and accidentally let the key presses occur simultaneously. Since such accidental simultaneous presses are usually followed by asynchronous releases of the two fingers, but finger pair chords are usually released synchronously, the decision whether the presses are asynchronous key taps or chord taps must be delayed until finger release can be checked for synchronization. In the meantime, step **610** places a hold on the keypress queue to prevent transmission of key symbols from the possible finger chord or any subsequent finger presses. To prevent long backups in key transmission, decision diamond **614** will eventually release the queue hold by having step **615** delete the synchronized presses from the keypress queue if both fingers remain touching a long time. Though this aborts the hypothesis that the presses were intended as key taps, the presses are also less likely to be key taps if the fingers are not lifted soon after touchdown.

[**0255**] If the synchronized fingers are not lifting, decision diamond **616** leaves the synchronization marker pending so synchronization checks can be continued with updated path parameters **600** after the next scan cycle. If the synchronized fingers are lifting, but decision diamond **618** finds with the help of the synchronization release search **603** that they are doing so asynchronously **618**, step **622** releases any holds on the keypress queue assuming any synchronized finger pair was intended to be two keypresses. Though the synchronized finger presses are not deleted from the keypress queue