EXHIBIT 3.12

The strain gage outputs of the Pointing Stick are conditioned by an IBM PC/Portable computer, equipped with a LabMaster A/D, D/A, and clock board. The computer makes resistance measurements on the pointing stick gages at 10 millisecond intervals, and emits a set of four pulse trains simulating standard Hawley Mouse signals, for speeds from 2 to about 10,000 pixels/second. Either these signals or signals from a standard mouse feed the PS/2 via an interface box (supplied by Microsoft during 1988-89) converting to serial PS/2 format. The experimental display is an IBM Type 8514 PS/2 color display, displaying 640 pixels horizontal and 350 vertical. Parameters are specified and results given in a coordinate system with 0,0 at screen center and $-1000 \le X \le 1000$, $-750 \le Y \le 750$, or approximately 0.14 mm per unit. Software in the PC/Portable allows full generality in generating, modifying, and applying transfer functions. The mouse is a Microsoft InPort(tm) Mouse purchased during 1989.

4. EXPERIMENTAL PROCEDURE

Two related experimental procedures were used. In both, Subject is seated before the computer display and keyboard in normal typing position, hands on the keys. Either the Pointing Stick or the mouse may be used; if the mouse, it is located adjacent to the keyboard on the preferred side, on a foam pad at about the level of the top of the keyboard. After signing in and entering experimental parameters, Subject initiates a trial by pressing a key ("t"). At the end of the trial, a score is presented, and the experimenter may choose to commence another trial, present an average score for the most recent group of trials, change experimental parameters, or terminate the experiment. The content of the 'trial' depends on the particular experiment.

1. Target Shooting. Subject selects targets presented as circles of random size and position on the screen. The situation being abstracted here is that of a user engaged in a typing task interspersed with single pointing actions; a pointing action begins and ends with the hands in typing home position. The 'trial' consists of 10 repetitions of the following: a blank screen is presented, with the mouse cursor (arrow) somewhere on it. Subject presses the J key (F if left-handed). The arrow appears at screen center, and a target outline appears at a random position on the screen. Subject moves to the pointing device, brings the arrow to point within the target, and presses a 'mouse-button' (on the mouse if a mouse is in use, the button below the space bar if the Pointing Stick). A hit (splash) or miss (beep) is signaled by the computer.

For a hit, the target and splash symbol remain on the screen until Subject returns to the keyboard and presses the J or F key again; for a miss, the screen blanks, ready for the next shot. For each shot, six items are recorded: target position (X,Y), target size, and three times: the time from initial keypress to first pointer movement, to 'hit', and to keyboard return. Misses are generally excluded from the data in analysis. Subject identification, experimental parameters, transfer function in use, date and time, and any other relevant conditions are also recorded in the same file.

The targets are circles of diameter randomly chosen from a uniform distribution between limits specified as an experimental parameter (usually 20 and 100 screen units, corresponding to the range from one character to a representative icon). Targets which extend beyond the screen edges, or are within one diameter of the center, are excluded.

2. Maze Running. A field of targets is presented which requires a sequence of pointings of varying directions and distances. Immediately upon the initiation of a trial, the screen is blanked and a field of numerals is presented, with the arrow in screen center. The object is to select the numerals in numerical order. Initially "1" is highlighted; as soon as it is selected by pressing the appropriate 'mouse' button with the arrow within the highlight, the highlight moves on to "2", and so on. For two-digit numerals, only the first digit is highlighted. Misses (inappropriate button presses) are disrewarded with a brief low-pitched sound, and counted. Each numeral must be successfully selected before the subject can proceed. An *event* begins with one successful selection (or the beginning of the trial) and ends with the next. The duration of each event is recorded. When the last numeral has been selected, the trial ends and the total elapsed time and number of errors are reported.

The same maze is used for a series of runs, so that in place of the random pointings of the other experiment, the maze presents a fixed sequence of pointings which is quickly learned. The targets are of fixed size, and, most important for mouse - Pointing Stick comparisons, the keyboard is not involved at all - this is a pure pointing task.

5. SUBJECTS

Subjects were 6 men ages 22-30 employed as co-op students at the T.J. Watson Research Center. All were experienced and proficient mouse users, but, aside from video game experience, naive to the Pointing Stick or any similar device. Subjects performed the experiments in random order, until scores

had settled (no significant difference between first and last 10 trials of a series of at least 40 trials (10 shots or 16 maze events per trial). Subjects reached different levels of proficiency, and comparisons are first within subject; those considered significant are consistent across subjects.

6. ANALYSIS

We can compare performance in our experiments in several ways. The simplest is to average measurements over enough trials for the target distance and size to average out. To see the effect of size or distance we can take a measurement as a function of the parameter in question, averaging out the other. Another option, used by previous authors [1] is to use the Fitts Difficulty Index, $DI = \log_2(D/S + .5)$, to collapse distance and size into a single parameter. We can then fit a line to the resulting point set, either before or after averaging points with similar *DI*, to obtain a two-parameter characterization and visualization of the data set, with a correlation coefficient to characterize the adequacy of the fit. This gave nice results, with correlation coefficient in the neighborhood of .98 for our larger data sets with averaging over intervals of 0.25 in *DI*.

For the maze experiment, total times are directly comparable between trials, and can be used as a sensitive measure of performance. To preserve the momentum of a sequence of pointing tasks, errors were tolerated in the maze experiment. In the target shooting experiment, to make all events directly comparable, we followed earlier workers [1] in dropping pointings in which errors occurred. One might question the effect of the different treatment of errors in the two experiments. When events in which errors occurred are eliminated from the maze data, the effect on the overall results is to increase the speed by perhaps 5%, without any qualitative change. Subjects were in part motivated by the scores which they saw at the end of each trial. In the maze the penalty for an error was loss of time, but more time might be lost in waiting to be sure of a hit before pressing the button. In the target experiment, errors did not directly affect the score, and it might be advantageous to deliberately miss a difficult target; we saw no suggestion that this occurred. The error rate was considerably higher in the maze experiment.

7. RESULTS

The velocity scale must be in a reasonable range - a control with a low top speed, or one which jumps uncontrollably at the slightest touch, is clearly unsatisfactory. The exact setting is less obvious. We repeatedly found that our intuition led to excessive sensitivity. The more interesting questions concern the shape of the transfer function, once the scaling is optimized.

In preliminary experiments we selected the following transfer functions for more careful characterization:

- Three linear functions with velocity scale factors respectively 1.5, .75 and .375. These are LIN1a, LIN1b and LIN1c.
- Two parabolic functions with velocity scale factors 1 and 2, called PAR1 and PAR2.
- Our current favorite shown above, 2Plateau. Its velocity scale factor of 1.5 puts the upper plateau of 2Plateau at 1120 pixels or about 50 cm per second.

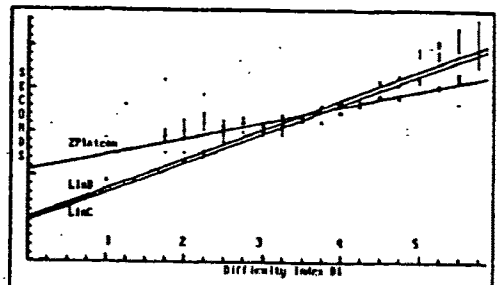


Figure 3. Comparison of 2Plateau with Linear Transfer Functions

Figure 3 is a plot of time against *DI* for 'target shooting' with 2Plateau, LIN1b, and LIN1c. The linear functions are faster at low difficulty (mainly distance - the range of target sizes in this experiment was 20 to 50 units). The simple numerical average times from keyboard to hit, for example, were 1.61, 1.71, and 1.65 seconds (average distances 645 ±1, sizes 35.5 ±0.5 for all three runs). Excluding points representing targets of size < 35 left the time against *DI* regression lines for 2Plateau and LIN1c essentially unchanged, but reduced the slope of the LIN1b line from .33 to .23. It appears that despite the small range of target sizes, the effect of size is significant.

The 'maze running' experiment gave a clearer distinction. Average run times and standard deviations in a sequence of runs, for one subject, were:

function	average time	S.D.	trials	slope t/*DI*
2Plateau	23.9	2.3	20	.30
LIN1b	27.9	2.4	30	.34
LIN1c	29.5	2.9	20	.51
LIN1a	27.8	2.4	30	.31
2Plateau	23.6	1.7	40	.30

LIN1a and LIN1b are not distinguished, but LIN1c differs from them at about 1 sigma, and all from 2Plateau at 2 sigma.

PAR1 and PAR2 gave performance similar to 2Plateau. Subjects reported objectionable fatigue using PAR1 and PAR2. The lower sensitivity could be compensated, but at the cost of physical effort - see discussion below. More sensitive parabolic functions were rejected in early screening as inadequately controllable for fine pointing.

Comparisons with sigmoid parabolic functions gave similar results - no significant differences in speed in either experiment, but noticeable differences in 'feel' and in fatigue effects.

8. POINTING STICK VERSUS MOUSE

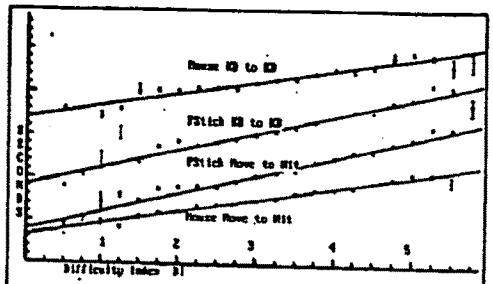


Figure 4. Mouse vs. Pointing Stick

Figure 4 shows the general result. The lower line fits the pointing time for the mouse, in the 'target shooting' experiment, taken from first movement (after 'homing') to selection of the target (hit). The upper line is the same run of the experiment, but timed from keyboard to keyboard (homing times included). The middle pair of lines gives the same information for the pointing stick. The averaged measurements for these runs are as follows:

	mouse	SD	Point	SD
Keyboard to first move	.64	.11	.39	.08
First move to hit	.76	.19	1.18	.35
Hit to keyboard	.72	.12	.09	.13
Keyboard to keyboard	2.12	.26	1.66	.39

Note that the time to reach the Pointing Stick is higher than expected, nearly 2/3 that for the mouse, despite the much shorter distance. The return time for the mouse is much longer than for the Pointing Stick.

The 'maze running' experiment, as a (nearly) pure pointing task, gives results very similar to the central part of the above experiment. The respective time against *DI* regression lines lie close to those for 'first move to hit' for both the mouse and the Pointing Stick. For most (but not all) subjects there was a significant delay between the hit on target n and the first move toward target n + 1, of the order of 0.1 second for the mouse and approaching twice this for the Pointing Stick. Best average times observed for the traversal of a sixteen point maze, starting at screen center, were 15.7 seconds, S.D. 1.8, 60 consecutive runs, for the mouse, and 20.0 seconds, S.D. 1.3, 120 consecutive runs, for the Pointing Stick.

9. DISCUSSION

In comparisons of mouse with Pointing Stick, it must be kept in mind that the subjects were highly experienced mouse users, but novices with the Pointing Stick. Therefore the comparisons can be used only as upper bounds on the differences to be expected in practice. Even so, for an isolated pointing action the Pointing Stick still has an advantage.

We have no firm explanation of the time from keyboard to first movement with the Pointing Stick, or of the difference in hit-to-first-move times in the maze between mouse and Pointing Stick. It is tempting to speculate that about 0.2 seconds is occupied in mental preparation for the move, that this is overlapped with the reaching action in the case of the mouse, and that the relative unfamiliarity of the Pointing Stick accounts for the longer time observed in the maze. The subject who exhibited very short hit-to-move times in the maze was using the LIN1 transfer functions, with slow cursor movement, and was observed to be 'shooting on the fly', never apparently stopping at a target; this strategy was not otherwise observed.

The relatively long return-to-keyboard time for the mouse is consistent with the fact that a key is a smaller target than the mouse.

The comparison of Pointing Stick transfer functions shows a wide range of subject adaptability in using strategies appropriate to the case in hand. For high sensitivity functions they automatically used intermittent contact with the stick, for low sensitivity they maintained contact and (in one case) adopted 'shoot-on-the-fly'. There may in fact be individual differences in optimum transfer function, although we have not observed this. In addition to the observed speed differences between linear and non-linear functions, differences of 'feel' and fatigue were observed, supporting our conjectures that at least two stable speeds, with an appropriate ratio between them, are desirable. The lower plateau of 2Plateau, at 1.5 cm/second, is appropriate for character-sized targets, but a bit fast for pixel targets, which would

be needed for a drawing application. While subjects could perform at speed with PAR1 and PAR2, the force required for long fast movements was too much to sustain for more than a few minutes of operation, while more sensitive functions made fine pointing too difficult.

We observed time/*DI* regression line slopes in the range of 0.12 (for the mouse) to .20±.03 for the Pointing Stick with optimal transfer function, and considerably higher with other functions. These contrast with apparently corresponding slopes of about 0.10 found previously [1, 3]. The latter effect is expected, for functions with low maximum speed - time increases linearly with distance, not logarithmically. For other functions, the explanation is presumably deeper, and requires further investigation.

10. CONCLUSIONS

We have been exploring alternative analogue pointing devices for computer interfaces. Laptop computers have no space for a mouse, and space is a problem in many office and other settings as well. The distraction and time of reaching for and returning from a mouse concerns us. We first considered adding sensors to a key under the index finger in a normal keyboard; signaling the use of the key for pointing or typing was distracting. We have placed joysticks in several keyboards and find the Pointing Stick between the G and H keys very useable. In experimenting with analogue pointing devices we have found the Pointing Stick can best the mouse in many situations.

For intermixed pointing and keyboard tasks the Pointing Stick is faster than the mouse. When three or more consecutive pointings occur the mouse can be up to 25% faster than the Pointing Stick. We note also that our Pointing Stick users' pointing speed continues to improve.

Our experience has been that users consistently over estimate their ability to control a fast pointing device. Reducing the rate of change for low speeds as in the parabolic, sigmoid parabolic and 2Plateau (Figure 1) functions increases subjects' speed for selecting small objects. The presence of two plateaus, with the proper ratio between them, makes precise control possible at relatively high sensitivity, greatly improving comfort and reducing fatigue. Adding the high speed tail of 2Plateau made users more comfortable with the Pointing Stick. Before this was added, two users literally bent the Pointing Stick (probably pressing over 5 pounds with their index fingers).

11. FUTURE DIRECTIONS

Following [5, 6] we have informally modeled pointing as a feedback control process, attempting to maintain what we think of as critical damping, which we find to yield the highest speed. A more critical treatment of this area should yield improvements in ease of use and in speed.

Other classes of force-to-motion functions are possible, in particular some degree of force-to-position mapping. Pure force-to-position mapping seems infeasible, but some mixed strategy, perhaps force-to-position locally with force-to-velocity at greater distances, should be worth investigating.

We informally measured how fast a subject could run our maze with his eyes; this was about 12 seconds or 3 seconds faster than the fastest pointing measured. Could this 25% speed difference be bridged? Could an eye tracking cursor positioner or "applications smart" transfer functions improve pointing speed?

BIBLIOGRAPHY

1. S. K. Card, W. K. English, and B. J. Burr. Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Test Keys for Text Selection on a CRT.. *Ergonomics*, 21(8):601-613, 1978..

2. Stuart K. Card, Thomas P. Moran, and Allen Newell. *Psychology of Human Computer Interaction*. Lawrence Erlbaum Associates, 1983.

3. P. M. Fitts. The Information Capacity of the Human Motor System In Controlling The Amplitude of Movement.. *Journal of Experimental Psychology*, 47:381-391, 1954..

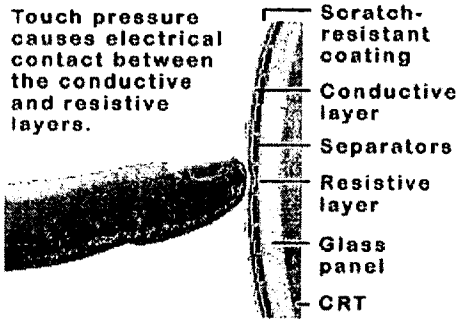
4. P. M. Fitts and J. R. Peterson. Information Capacity of Discrete Motor Responses. *Journal of Experimental Psychology*, 67(2):103-112, 1964..

5. T. O. Kvalseth. Information Capacity of Two-Dimensional Human Motor Responses. *Ergonomics*, 24(7):573-575, 1981..

6. D. E. Meyer, S. Kornblum, R. A. Abrams, C. E. Wright, and J. E. K. Smith. Optimality in Human Motor-Performance - Ideal Control of Rapid Aimed Movements. *Psychological Review*, 95(3):340-370, 1988..

TouchScreens.com

4-Wire Resistive Touchscreens



4-Wire Resistive touch technology consists of a glass or acrylic panel that is coated with electrically condictive and resistive layers. The thin layers are separated by invisible separator dots. When operating, an electrical current moves through the screen. When pressure is applied to the screen the layers are pressed together, causing a change in the electrical current and a touch event to be registered.

4-Wire Resistive type touch screens are generally the most affordable. Although clarity is less than with other touch screen types, resistive screens are very durable and can be used in a variety of environments. This type of screen is recommended for individual, home, school, or office use, or less demanding point-of-sale systems, restaurant systems, etc.

Advantages

- High touch resolution
- Pressure sensitive, works with any stylus
- Not affected by dirt, dust, water, or light
- Affordable touchscreen technology

Disadvantages

- 75 % clarity
- Resistive layers can be damaged by a sharp object
- Less durable then 5-Wire Resistive technology

Touchscreen Specifications

Touch Type:	4-Wire Resistive
Screen Sizes:	12"-20" Diagonal
Cable Interface:	PC Serial/COM Port or USB Port
Touch Resolution:	1024 x 1024
Response Time:	10 ms. maximum
Activation Force:	50-120 grams per square centimer
Positional Accuracy:	3mm maximum error
Light Transmission:	80% nominal
Light Transmission:	80% nominal
Scratch Resistance:	3H pencil hardness
Life Expectancy:	3 million touches at one point
Temperature:	Operating: -10°C to 70°C Storage: -30°C to 85°C

Humidity:	Pass 40 degrees C, 95% RH for 96 hours.
Chemical Resistance:	Alcohol, acetone, grease, and general household detergent
Software Drivers:	Windows XP / 2000 / NT / ME / 98 / 95, Linux, Macintosh OS

TouchScreens.com is owned and operated by Mass Multimedia, Inc. Call: 1-800-348-8610 E-mail: info@touchscreens.com

Applications for a Switched-Capacitor Instrumentation Building Block

Jim Williams

CMOS analog IC design is largely based on manipulation of charge. Switches and capacitors are the elements used to control and distribute the charge. Monolithic filters, data converters and voltage converters rely on the excellent characteristics of IC CMOS switches. Because of the importance of switches in their circuits, CMOS designers have developed techniques to minimize switch induced errors, particularly those associated with stray capacitance and switch timing. Until now, these techniques have been used only in the internal construction of monolithic devices. A new device, the LTC1043, makes these switches available for board level use. Multi-pole switching and a self-driven, non-overlapping clock allow the device to be used in circuits which are impractical with other switches.

Conceptually, the LTC1043 is simple. Figure 1 details its features. The oscillator, free-running at 200kHz, drives a non-overlapping clock. Placing a capacitor from pin 16 to ground shifts the oscillator frequency downward to any desired point. The pin may also be driven from an external source, synchronizing the switches to external circuitry. A non-overlapping clock controls both DPDT switch sections. The non-overlapping drive prevents simultaneous conduction in the series connected switch sections.

Charge balancing circuitry cancels the effects of stray capacitance. Pins 1 and 10 may be used as guard points for pins 3 and 12 in particularly sensitive applications.

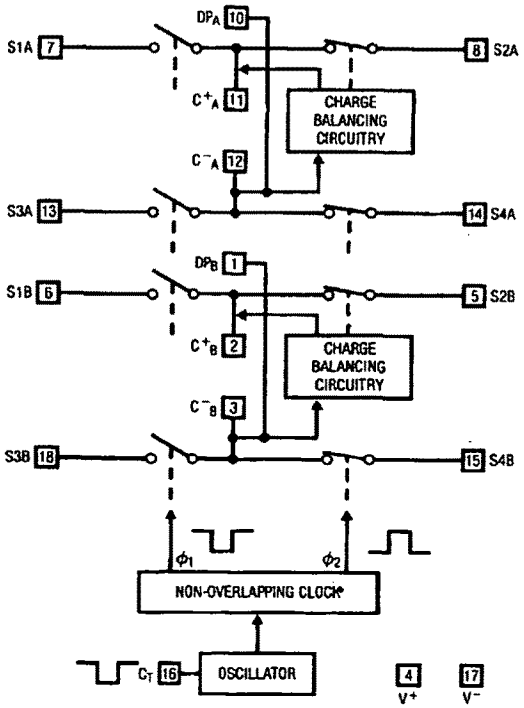


Figure 1. Block Diagram of LTC1043 Showing Individual Switches

Although the device's operation is simple, it permits surprisingly sophisticated circuit functions. Additionally, the careful attention paid to switching characteristics makes implementing such functions relatively easy. Discrete timing and charge-balance compensation networks are eliminated, reducing component count and trimming requirements.

Classical analog circuits work by utilizing continuous functions. Their operation is usually described in terms of voltage and current. Switched-capacitor based circuits are sampled data systems which approximate continuous functions with bandwidth limited by the sampling frequency. Their operation is described in the distribution of charge over time. To best understand the circuits which follow, this distinction should be kept in mind. Analog sampled data and carrier based systems are less common than true continuous approaches, and developing a working familiarity with them requires some thought.

Switched-capacitor approaches have greatly aided analog MOS IC design. The LTC1043 brings many of the freedoms and advantages of CMOS IC switched-capacitor circuits to the board level, providing a valuable addition to available design techniques.

Instrumentation Amplifier

Figure 2 uses the LTC1043 to build a simple, precise instrumentation amplifier. An LTC1043 and an LT1013 dual op amp are used, allowing a dual instrumentation amplifier using just two packages. A single DPDT section converts the differential input to a ground referred single-ended signal at the LT1013's input. With the input switches closed, C1 acquires the input signal. When the input switches open, C2's switches close and C2 receives charge. Continuous clocking forces C2's voltage to equal the difference between the circuit's inputs. The 0.01μF capacitor at pin 16 sets the switching frequency at 500Hz. Common-mode voltages are rejected by over 120dB and drift is low.

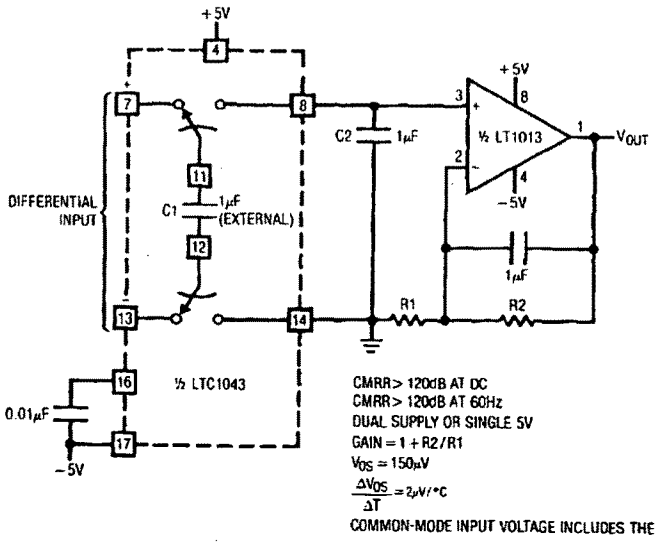


Figure 2. ±5V Precision Instrumentation Amplifier

Amplifier gain is set in the conventional manner. This circuit is a simple, economical way to build a high performance instrumentation amplifier. Its DC characteristics rival any IC or hybrid unit and it can operate from a single 5V supply. The common-mode range includes the supply rails, allowing the circuit to read across shunts in the supply lines. The performance of the instrumentation amplifier depends on the output amplifier used. Specifications for an LT1013 appear in the figure. Lower figures for offset, drift and bias current are achievable by employing type LT1001, LT1012, LT1056 or the chopper-stabilized LTC1052.

Ultra-High Performance Instrumentation Amplifier

Figure 3 is similar to Figure 2, but utilizes the remaining LTC1043 section to construct a low drift chopper amplifier. This approach maintains the true differential inputs while achieving 0.1μV/°C drift. The differential input is converted to a single-ended potential at pin 7 of the LTC1043. This voltage is chopped into a 500Hz square wave by the switching action of pins 7, 11, and 8. A1, AC coupled, amplifies this signal. A1's square wave output, also AC coupled, is synchronously demodulated by switches 12, 14, and 13. Because this switch section is synchronously driven with the input chopper, proper amplitude and polarity information is presented to A2, the DC output amplifier. This stage integrates the square wave into a DC voltage to provide the output. The output is divided down and fed back to pin 8 of the input chopper where it serves as the zero signal reference. Because the main amplifier is AC coupled, its DC terms do not affect overall circuit offset, resulting in the extremely low offset and drift noted in the specifications. This circuit offers lower offset and drift than any commercially available instrumentation amplifier.

Lock-In Amplifier

The AC carrier approach used in Figure 3 may be extended to form a "lock-in" amplifier. A lock-in amplifier works by synchronously detecting the carrier modulated output of the signal source. Because the desired signal information is contained within the carrier, the system constitutes an extremely narrow-band amplifier. Non-carrier related components are rejected and the amplifier passes only signals which are coherent with the carrier. In practice, lock-in amplifiers can extract a signal 120dB below the noise level.

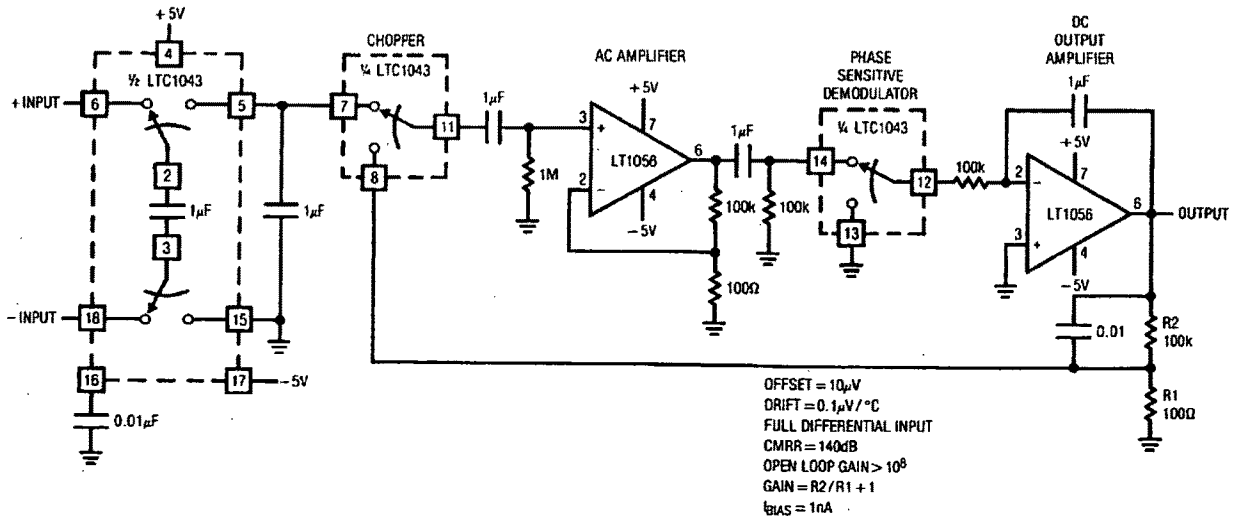


Figure 3. Chopper-Stabilized Instrumentation Amplifier

Figure 4 shows a lock-in amplifier which uses a single LTC1043 section. In this application, the signal source is a thermistor bridge which detects extremely small temperature shifts in a biochemical microcalorimetry reaction chamber.

The 500Hz carrier is applied at T1's input (Trace A, Figure 5). T1's floating output drives the thermistor bridge, which presents a single-ended output to A1. A1 operates at an AC gain of 1000. A 60Hz broadband noise source is also deliberately injected into A1's input (Trace B). The carrier's zero crossings are detected by C1. C1's output clocks the LTC1043 (Trace C). A1's output (Trace D) shows the desired 500Hz signal buried within the 60Hz noise source. The LTC1043's zero-cross-synchronized switching at A2's positive input (Trace E) causes A2's gain to alternate between plus and minus one. As a result, A1's output is synchronously demodulated by A2. A2's output (Trace F) consists of demodulated carrier signal and non-coherent components. The desired carrier amplitude and polarity information is discernible in A2's output and is extracted by filter-averaging at A3. To trim this circuit, adjust the phase potentiometer so that C1 switches when the carrier crosses through zero.

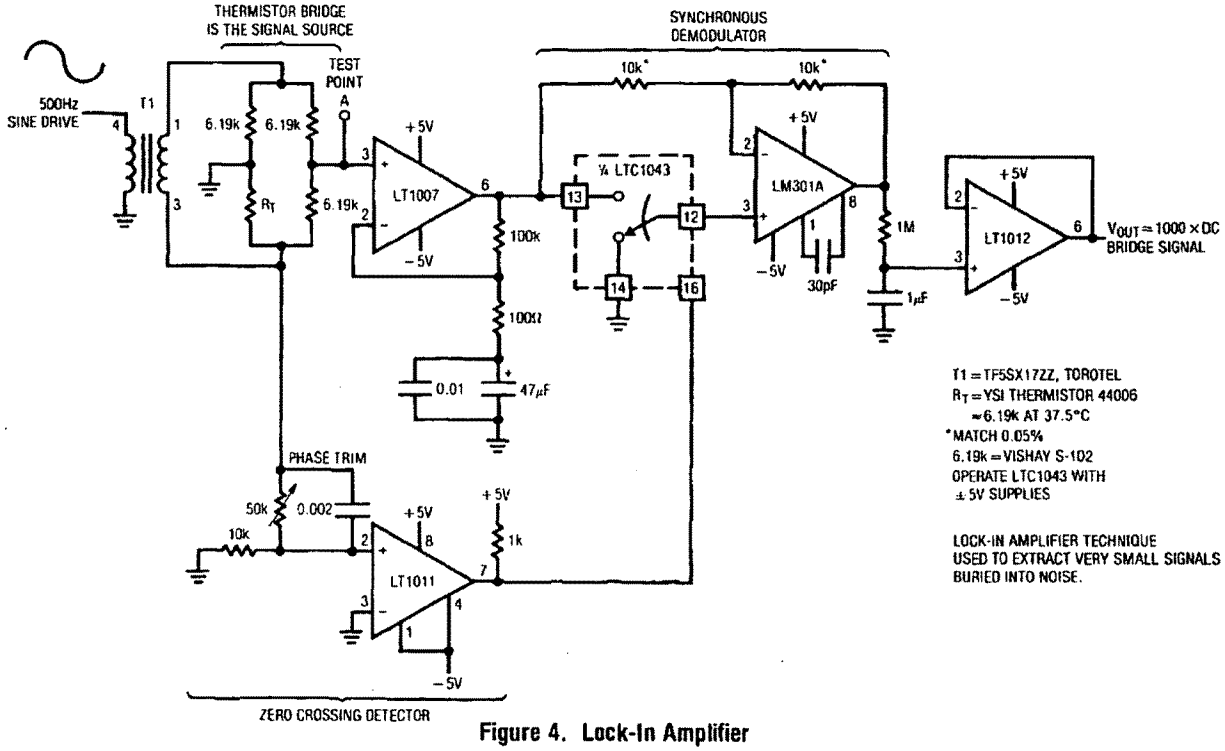


Figure 4. Lock-In Amplifier

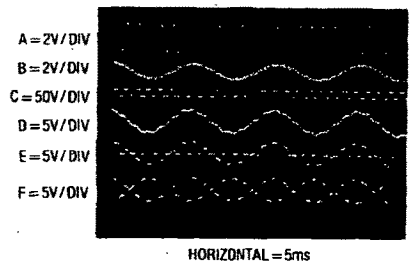


Figure 5

Wide Range, Digitally Controlled, Variable Gain Amplifier

Aside from low drift and noise rejection, another dimension in amplifier design is variable gain. Designing a wide range, digitally variable gain block with good DC stability is a difficult task. Such configurations usually involve relays or temperature compensated FET networks in expensive and complex arrangements. The circuit shown in Figure 6 uses the LTC1043 in a variable gain amplifier which features continuously variable gain from 0–1000, gain stability of 20ppm/°C and single-ended or differential input. The circuit uses two separate LTC1043s. Unit A is clocked by a frequency input which could be derived from a host processor. LTC1043B is continuously clocked by a 1kHz source which could also be processor supplied. Both LTC1043s function as the sampled data equivalent of a resistor within the bandwidth set by A1's 0.01μF value and the switched-capacitor equivalent feedback resistor. The time-averaged current delivered to the summing point by LTC1043A is a function of the 0.01μF capacitor's input-derived voltage and the commutation frequency at pin 16. Low commutation frequencies result in small time-averaged current values, approximating a large input resistor. Higher frequencies produce an equivalent small input resistor. LTC1043B, in A1's feedback path, acts in a similar fashion. For the circuit values given, the gain is simply:

$$G = \frac{f_{IN}}{10} \times \frac{0.01\mu F}{100pF}$$

Gain stability depends on the ratiometric stability between the 1kHz and variable clocks (which could be derived from a common source) and the ratio stability of the capacitors. For polystyrene types, this will typically be 20ppm/°C. The circuit input, determined by the pin connections shown in the figure, may be either single-ended or fully differential. Additionally, although A1 is connected as an inverter, the circuit's overall transfer function may be either positive or negative. As shown, with pins 13A and 7A grounded and the input applied to 8A, it is negative.

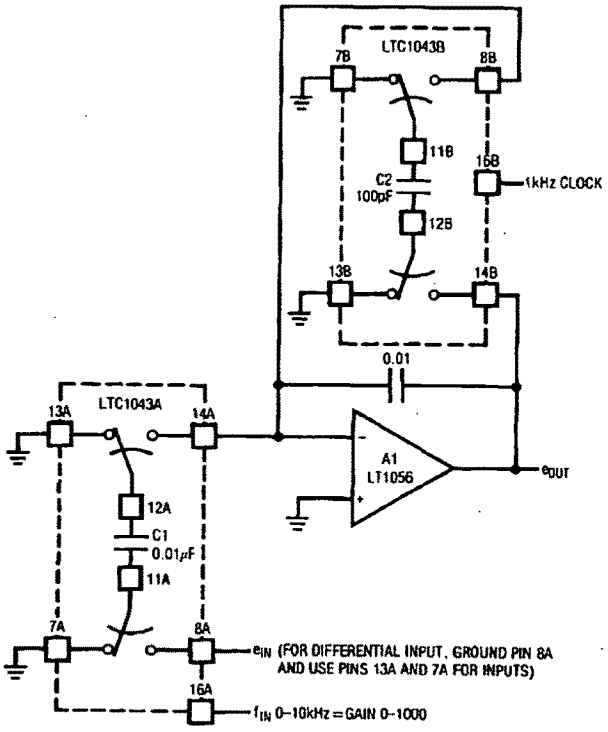


Figure 6. Variable Gain Amplifier

Precision, Linearized Platinum RTD Signal Conditioner

Figure 7 shows a circuit which provides complete, linearized signal conditioning for a platinum RTD. One side of the RTD sensor is grounded, often desirable for noise considerations. This LTC1043 based circuit is considerably simpler than instrumentation or multi-amplifier based designs and will operate from a single 5V supply. A1 serves as a voltage-controlled ground referred current source by differentially sensing the voltage across the 887Ω feedback resistor. The LTC1043 section which does this presents a single-ended signal to A1's negative input, closing a loop. The 2k-0.1μF combination sets amplifier roll-off well below the LTC1043's switching frequency and the configuration is stable. Because A1's loop forces a fixed voltage across the 887Ω resistor, the current through R_P is constant. A1's operating point is primarily fixed by the 2.5V LT1009 voltage reference. The RTD's constant current forces the voltage across it to vary with its resistance, which has a nearly linear positive temperature coefficient. The non-linearity could cause several degrees of error over the circuit's 0°C–400°C operating range. A2 amplifies R_P's output, while simultaneously supplying non-linearity correction. The correction is implemented by feeding a portion of A2's output back to A1's input via the 10k–250k divider. This causes the current supplied to R_P to slightly shift with its operating point, compensating sensor non-linearity to within ±0.05°C. The remaining LTC1043 section furnishes A2 with a differential input. This allows an offsetting potential, derived from the LT1009 reference, to be subtracted from R_P's output. Scaling is arranged so 0°C equals 0V at A2's output. Circuit gain is set by A2's feedback values and linearity correction is derived from the output.

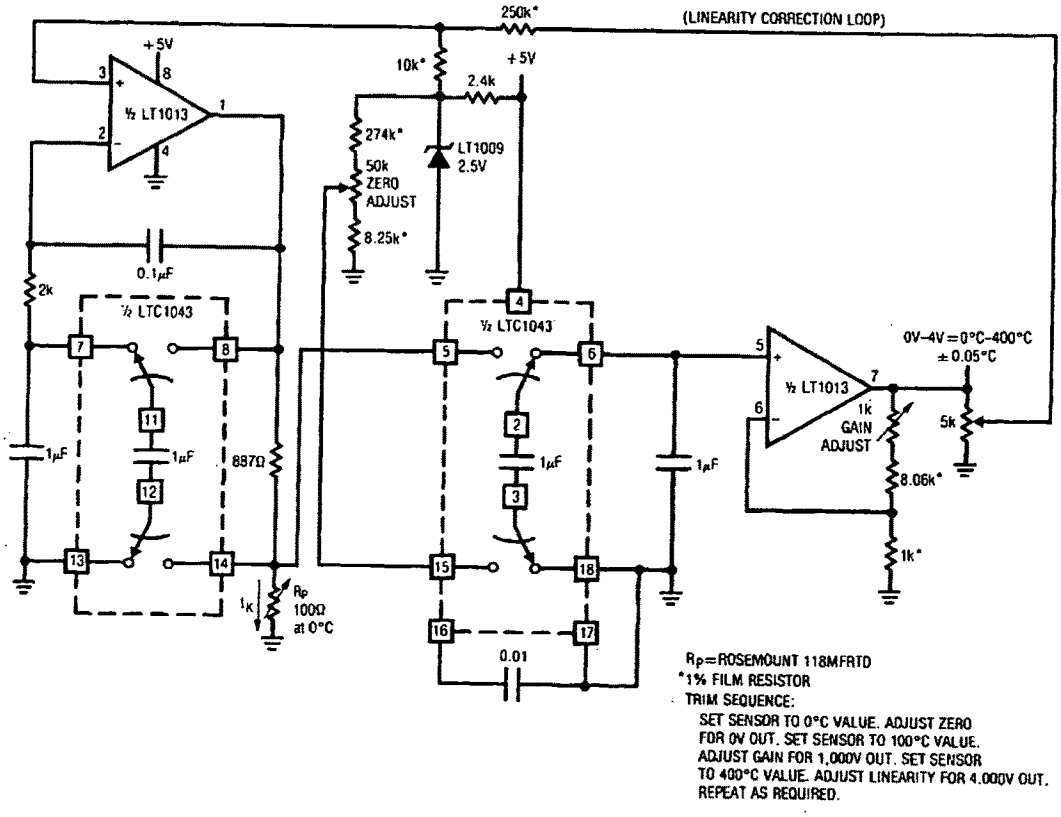


Figure 7. Linearized Platinum Signal Conditioner

To calibrate this circuit, substitute a precision decade box (e.g., General Radio 1432k) for R_P. Set the box to the 0°C value (100.00Ω) and adjust the offset trim for a 0.00V output. Next, set the decade box for a 140°C output (154.26Ω) and adjust the gain trim for a 1.400V output reading. Finally, set the box to 249.0°C (400.00°C) and trim the linearity adjustment for a 4.000V output. Repeat this sequence until all three points are fixed. Total error over the entire range will be within ±0.05°C. The resistance values given are for a nominal 100.00Ω (0°C) sensor. Sensors deviating from this nominal value can be used by factoring in the deviation from 100.00Ω. This deviation, which is manufacturer specified for each individual sensor, is an offset term due to winding tolerances during fabrication of the RTD. The gain slope of the platinum is primarily fixed by the purity of the material and is a very small error term.

Note that A1 constitutes a voltage controlled current source with input and output referred to ground. This is a difficult function to achieve and is worthy of separate mention.

Relative Humidity Sensor Signal Conditioner

Relative humidity is a difficult physical parameter to transduce, and most transducers available require fairly complex signal conditioning circuitry. Figure 8 combines two LTC1043s with a recently introduced capacitively based humidity transducer in a simple charge-pump based circuit.

The sensor specified has a nominal 500pF capacitance at RH = 76%, with a slope of 1.7pF/% RH. The average voltage across this device must be zero. This provision prevents deleterious electrochemical migration in the sensor. LTC1043A inverts a resistively scaled portion of the LT1009 reference, generating a negative potential at pin 14A. LTC1043B alternately charges and discharges the humidity sensor via pins 12B, 13B, and 14B. With 14B and 12B connected, the sensor charges via the 1μF unit to the negative potential at pin 14A. When the 14B-12B pair opens, 12B is connected to A1's summing point via 13B. The sensor now discharges into the summing point through the 1μF capacitor. Since the charge

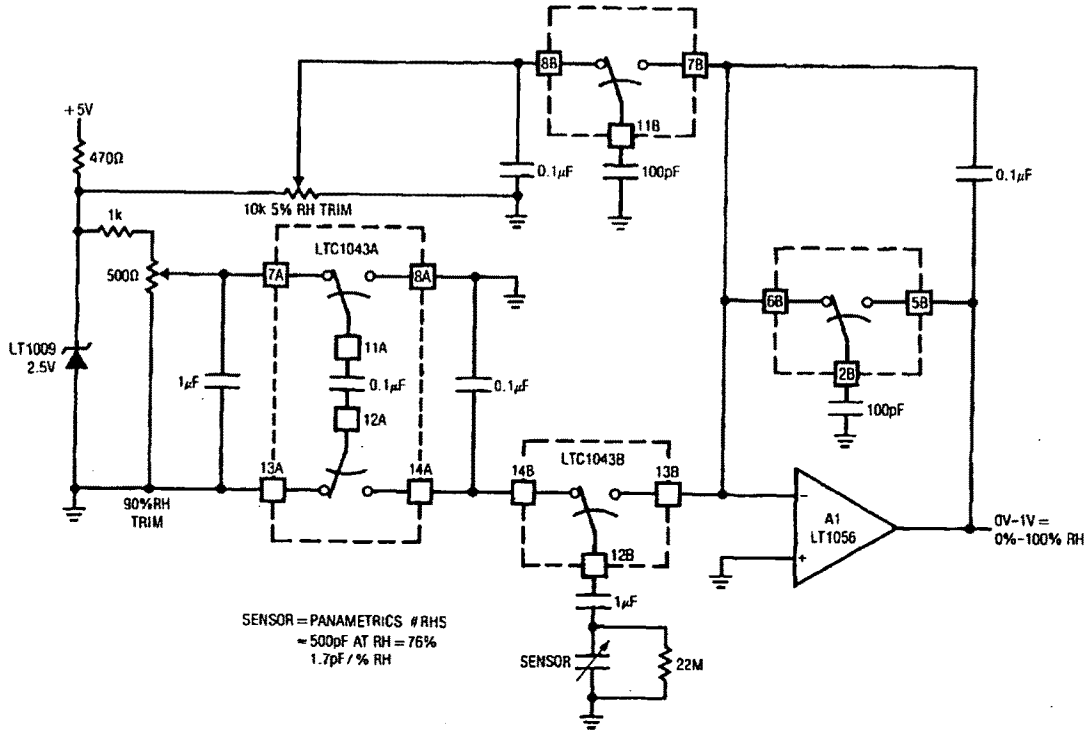


Figure 8. Relative Humidity Signal Conditioner

voltage is fixed, the average current into the summing point is determined by the sensor's humidity related value. The 1μF value AC couples the sensor to the charge-discharge path, maintaining the required zero average voltage across the device. The 22M resistor prevents accumulation of charge, which would stop current flow. The average current into A1's summing point is balanced by packets of charge delivered by the switched-capacitor network in A1's feedback loop. The 0.1μF capacitor gives A1 an integrator-like response, and its output is DC.

To allow 0% RH to equal 0V, offsetting is required. The signal and feedback terms biasing the summing point are expressed in charge form. Because of this, the offset must also be delivered to the summing point as charge, instead of a simple DC current. If this is not done, the circuit will be affected by frequency drift of LTC1043B's oscillator. Section 8B-11B-7B serves this function, delivering LT1009-referenced offsetting charge to A1. Drift terms in this circuit include the LT1009 and the ratio stability of the sensor and the 100pF capacitors. These terms are well within the sensor's 2% accuracy specification and temperature compensation is not required. To calibrate this circuit, place the sensor in a known 5% RH environment and adjust the "5% RH trim" for 0.05V output. Next, place the sensor in a 90% RH environment and set the "90% RH trim" for 900mV output. Repeat this procedure until both points are fixed. Once calibrated, this circuit is accurate within 2% in the 5%–90% RH range.

Figure 9 shows an alternate circuit which requires two op amps but needs only one LTC1043 package. This circuit retains insensitivity to clock frequency while permitting a DC offset trim. This is accomplished by summing in the offset current after A1.

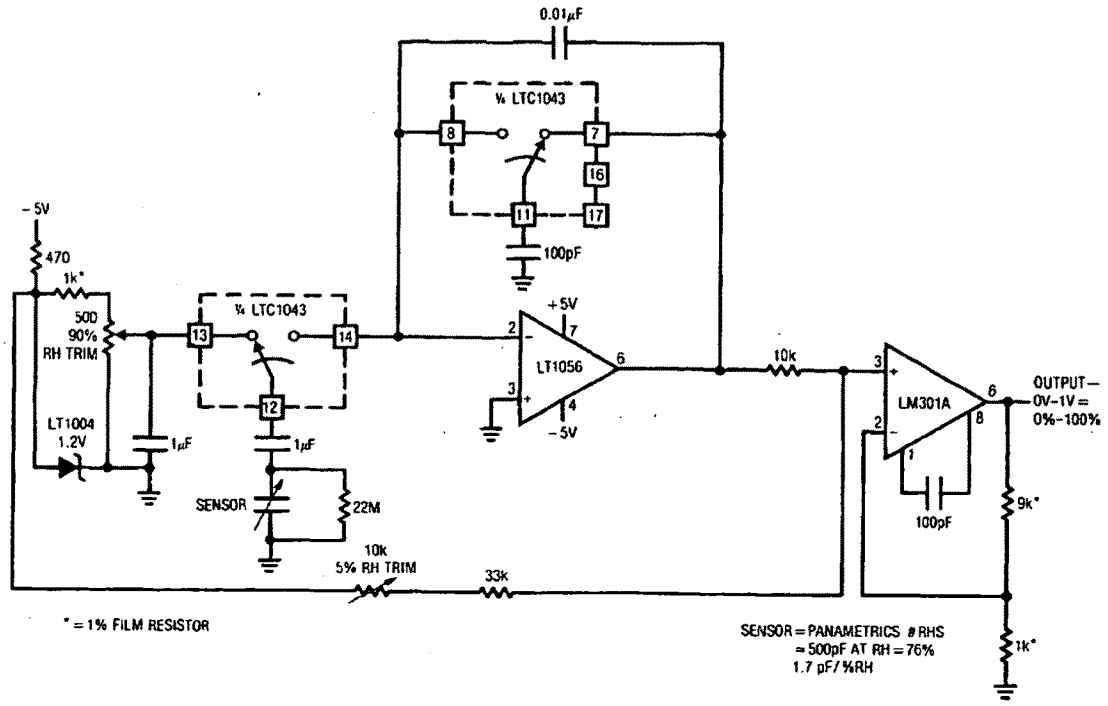


Figure 9. Relative Humidity Signal Conditioner

LVDT Signal Conditioner

LVDTs (Linear Variable Differential Transformers) are another example of a transducer which the LTC1043 can signal condition. An LVDT is a transformer with a mechanically actuated core. The primary is driven by a sine wave, usually amplitude stabilized. Sine drive eliminates error inducing harmonics in the transformer. The two secondaries are connected in opposed phase. When the core is positioned in the magnetic center of the transformer, the secondary outputs cancel and there is no output. Moving the core away from the center position unbalances the flux ratio between the secondaries, developing an output. Figure 10 shows an LTC1043 based LVDT signal conditioner. A1 and its associated components furnish the amplitude stable sine wave source. A1's positive feedback path is a Wein bridge, tuned for 1.5kHz. Q1, the LT1004 reference, and additional components in A1's negative loop unity-gain stabilize the amplifier. A1's output (Trace A, Figure 11), an amplitude stable sine wave, drives the LVDT. C1 detects zero crossings and feeds the LTC1043 clock pin (Trace B). A speed-up network at C1's input compensates LVDT phase shift, synchronizing the LTC1043's clock to the transformer's output zero crossings. The LTC1043 alternately connects each end of the

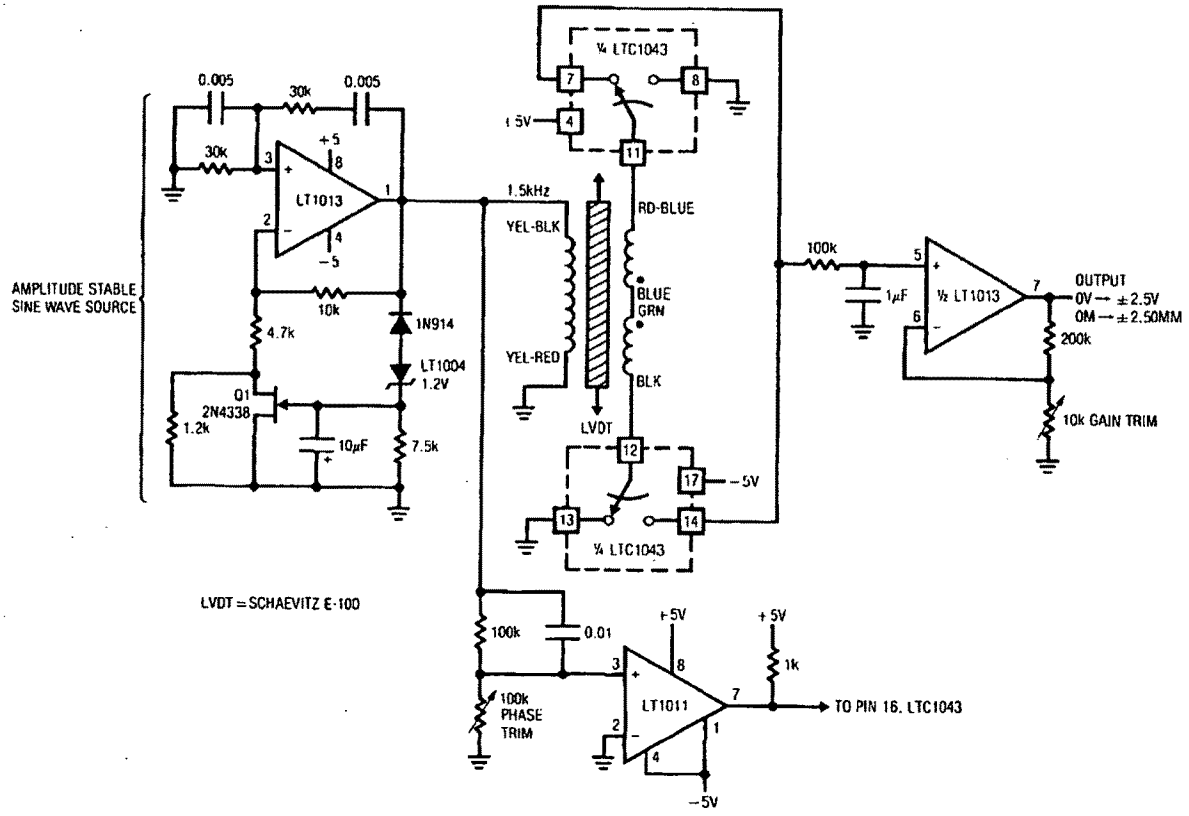


Figure 10. LVDT Signal Conditioner

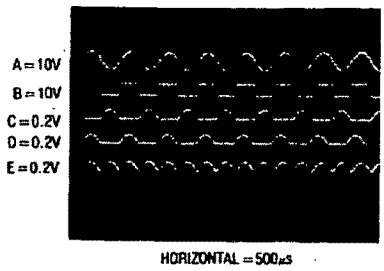


Figure 11

transformer to ground, resulting in positive half-wave rectification at pins 7 and 14 (Traces C and D, respectively). These points are summed (Trace E) at a low pass filter which feeds A2. A2 furnishes gain scaling and the circuit's output.

The LTC1043's synchronized clocking means the information presented to the low pass filter is amplitude and phase sensitive. The circuit output indicates how far the core is from center and on which side.

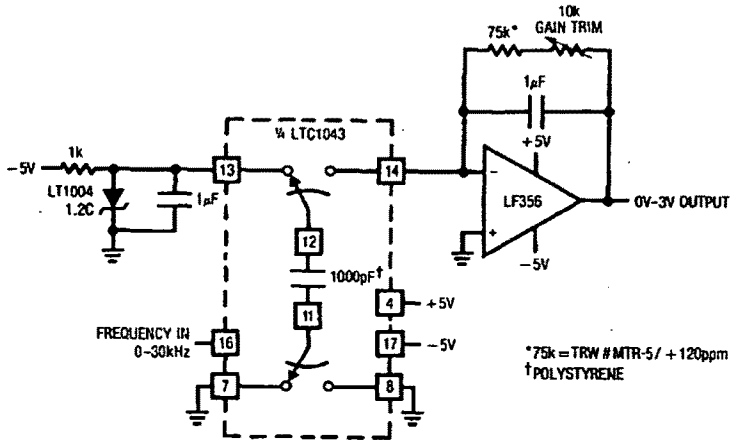
To calibrate this circuit, center the LVDT core in the transformer and adjust the phase trim for 0V output. Next, move the core to either extreme position and set the gain trim for 2.50V output.

Charge Pump F→V and V→F Converters

Figure 12 shows two related circuits, both of which show how the LTC1043 can simplify a precision circuit function. Charge pump F→V and V→F converters usually require substantial compensation for non-ideal charge gating behavior. These examples equal the performance of such circuits, while requiring no compensations. These circuits are economical, component count is low, and the 0.005% transfer linearity equals that of more complex designs. Figure 12A is an F→V converter. The LTC1043's clock pin is driven from the input (Trace A, Figure 13). With the input high, pins 12 and 13 are shorted and 14 is open. The 1000pF capacitor receives charge from the 1μF unit, which is biased by the LT1004. At the input's negative-going edge, pins 12–13 open and 12–14 close. The 1000pF capacitor quickly removes current (Trace B) from A1's summing node. Initially, current is transferred through A1's feedback capacitor and the amplifier output goes negative (Trace C). When A1 recovers, it slews positive to a level which resets the summing junction to zero. A1's 1μF feedback capacitor averages this action over many cycles and the circuit output is a DC level linearly related to frequency. A1's feedback resistors set the circuit's DC gain. To trim the circuit, apply 30kHz in and set the 10kΩ gain trim for exactly 3V output. The primary drift term in this circuit is the 120ppm/°C tempco of the 1000pF capacitor, which should be polystyrene. This can be reduced to within 20ppm/°C by using a feedback resistor with an opposing tempco (e.g., TRW #MTR-5/+120ppm). The input pulse width must be low for at least 100ns to allow complete discharge of the 1000pF capacitor.

In Figure 12B, the LTC1043 based charge pump is placed in A1's feedback loop, resulting in a V→F converter. The clock pin is driven from A1's output. Assume that A1's negative input is just below 0V. The amplifier output is positive. Under these conditions, LTC1043's pins 12 and 13 are shorted and 14 is open, allowing the 0.01μF capacitor to charge toward the negative 1.2V LT1004. When the input-voltage-derived current forces A1's summing point (Trace A, Figure 13) positive, its output (Trace B) goes negative. This reverses the LTC1043's switch states, connecting pins 12 and 14. Current flows from the summing point into the 0.01μF capacitor (Trace C). The 30pF-22k combination at A1's positive input (Trace D) ensures A1 will remain low long enough for the 0.01μF capacitor to completely reset to zero. When the 30pF-22k positive feedback path decays, A1's output returns positive and the entire cycle repeats. The oscillation frequency of this action is directly related to the input voltage with a transfer linearity of 0.005%.

Start-up or overdrive conditions could force A1 to go to the negative rail and stay there. Q1 prevents this by pulling the summing point negative if A1's output stays low long enough to charge the 1μF-330k RC. Two LTC1043 switch sections provide complementary sink-source outputs. Similar to the F→V circuit, the 0.01μF capacitor is the primary drift term, and the resistor type noted above will provide optimum tempco cancellation. To calibrate this circuit, apply 3V and adjust the gain trim for a 30kHz output.



A. Frequency to Voltage Converter

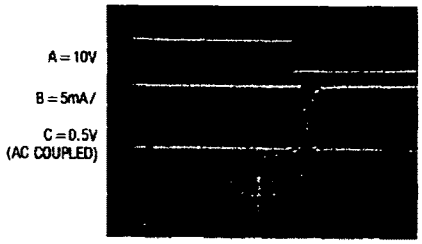
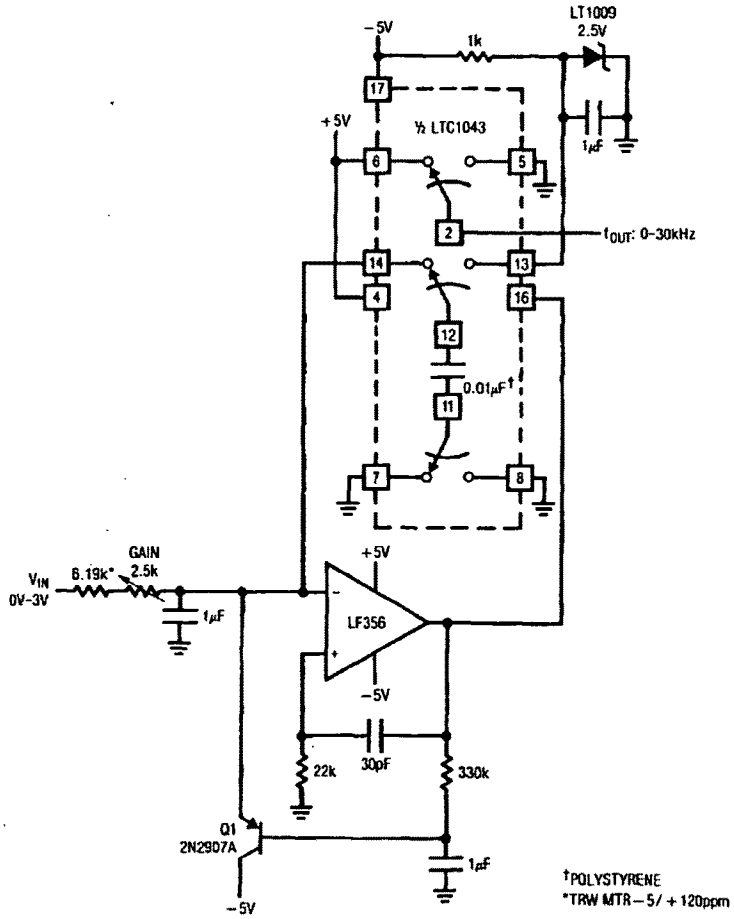


Figure 13



B. Voltage to Frequency Converter

Figure 12

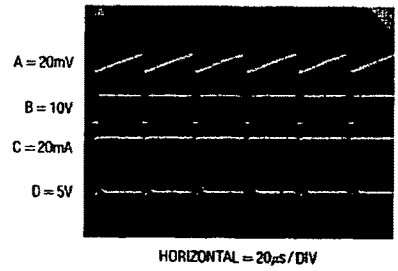


Figure 14

12-Bit A→D Converter

Figure 15 shows the LTC1043 used to implement an economical 12-bit A→D converter. The circuit is self-clocking, has a serial output, and completes a full-scale conversion in 25ms.

Two LTC1043s are used in this design. Unit A free-runs, alternately charging the 100pF capacitor from the LT1004 reference source and then dumping it into A1's summing point. A1, connected as an integrator, responds with a linear ramp output (Trace B, Figure 16). This ramp is compared to the input voltage by C1B. When the crossing occurs, C1B's output goes low (Trace C, just faintly visible in the photograph), setting the flip-flop high (Trace D). This pulls LTC1043's pin 16 high, resetting A1's integrator

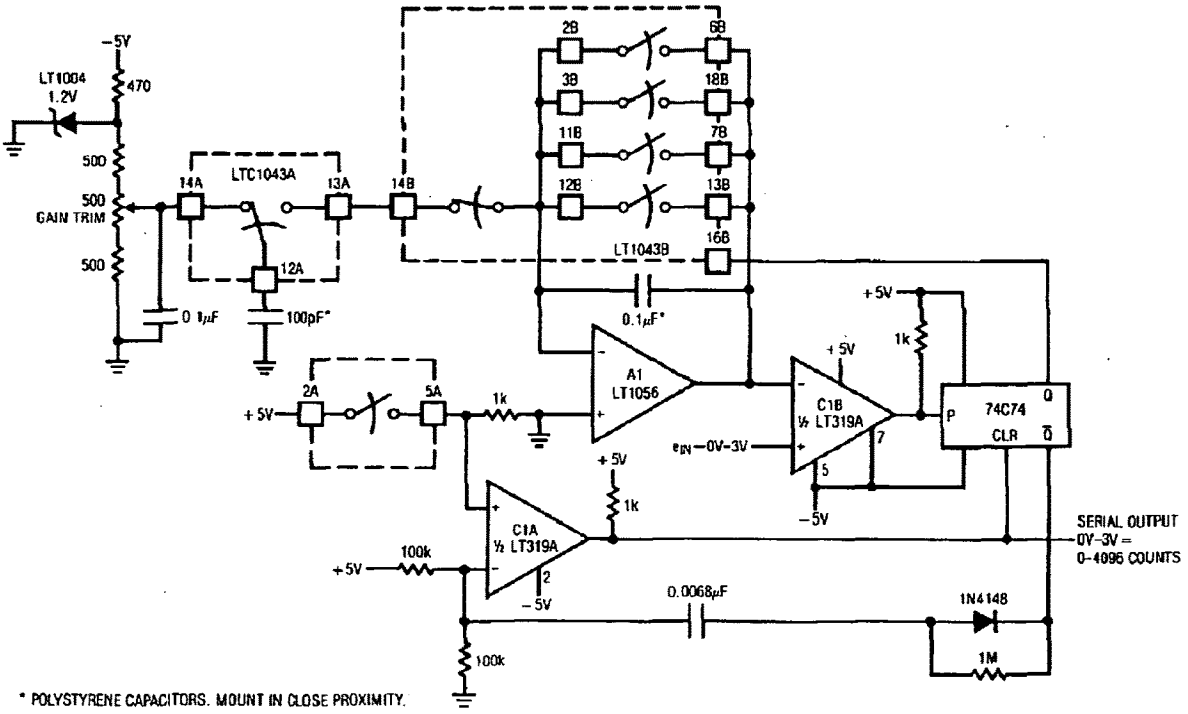


Figure 15. 12-Bit A→D Converter

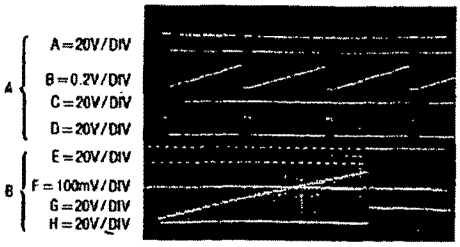


Figure 16

capacitor via the paralleled switches. Simultaneously, pin 14B opens, preventing charge from being delivered to A1's summing point during the reset. The flip-flop's Q output, low during this interval, causes an AC negative-going spike at C1A. This forces C1A's output high, inserting a gap in the output clock pulse stream (Trace A). The width of this gap, set by the components at C1A's negative input, is sufficient to allow a complete reset of A1's integrating capacitor. The number of pulses between gaps is directly related to the input voltage. The actual conversion begins at the gap's negative edge and ends at its positive edge. The flip-flop output may be used for resetting. Alternately, a processor driven "time-out" routine can determine the end of conversion. Traces E through H offer expanded scale versions of Traces A through D, respectively. The staircase detail of A1's ramp output reflects the charge pumping action at its summing point. Note that drift in the 100pF and 0.1μF capacitors, which should be polystyrene, ratiometrically cancels. Full-scale drift for this circuit is typically 20ppm/°C, allowing it to hold 12-bit accuracy over 25°C + 10°C. To calibrate the circuit, apply 3V in and trim the gain potentiometer for 4096 pulses out between data stream gaps.

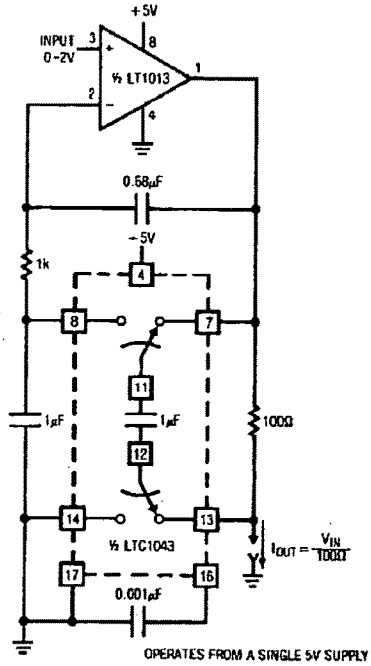


Figure 17. Voltage Controlled Current Source with Ground Referred Input and Output

Miscellaneous Circuits

Figures 17–22 show a group of miscellaneous circuits, most of which are derivations of applications covered in the text. As such, only brief comments are provided.

Voltage-Controlled Current Source—Grounded Source and Load

This is a simple, precise voltage-controlled current source. Bipolar supplies will permit bipolar output. Configurations featuring a grounded voltage control source and a grounded load are usually more complex and depend upon several components for stability. In this circuit, accuracy and stability are almost entirely dependent on the 100Ω shunt.

Current Sensing in Supply Rails

The LTC1043 can sense current through a shunt in either of its supply rails (Figure 18). This capability has wide application in battery and solar-powered systems. If the ground-referred voltage output is unloaded by an amplifier, the shunt can operate with very little voltage drop across it, minimizing losses.

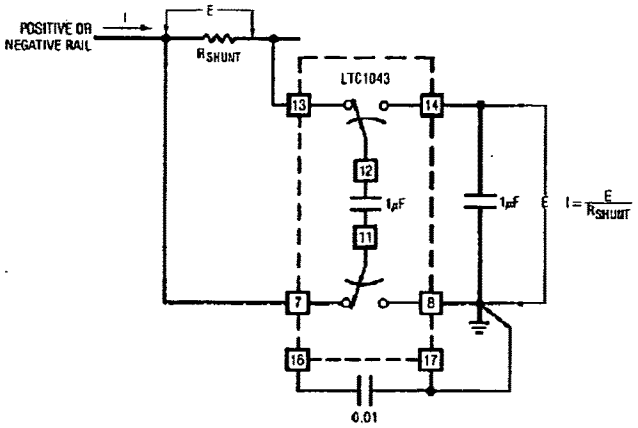


Figure 18. Precision Current Sensing in Supply Rails

0.01% Analog Multiplier

Figure 19, using the V→F and F→V circuits previously described, forms a high precision analog multiplier. The F→V input frequency is locked to the V→F output because the LTC1043's clock is common to both sections. The F→V's reference is used as one input of the multiplier, while the V→F furnishes the other. To calibrate, short the X and Y inputs to 1.7320V and trim for a 3V output.

Inverting a Reference

Figure 20 allows a reference to be inverted with 1ppm accuracy. This circuit features high input impedance and requires no trimming.

Low Power, 5V Driven, Temperature Compensated Crystal Oscillator

Figure 21 uses the LTC1043 to differentiate between a temperature sensing network and a DC reference. The single-ended output biases a varactor tuned crystal oscillator to compensate drift. The varactor-crystal network has high DC impedance, eliminating the need for an LTC1043 output amplifier.

Simple Thermometer

Figure 22's circuit is conceptually similar to the platinum RTD example of Figure 7. The thermistor network specified eliminates the requirement for a linearity trim, at the expense of accuracy and range of operation.

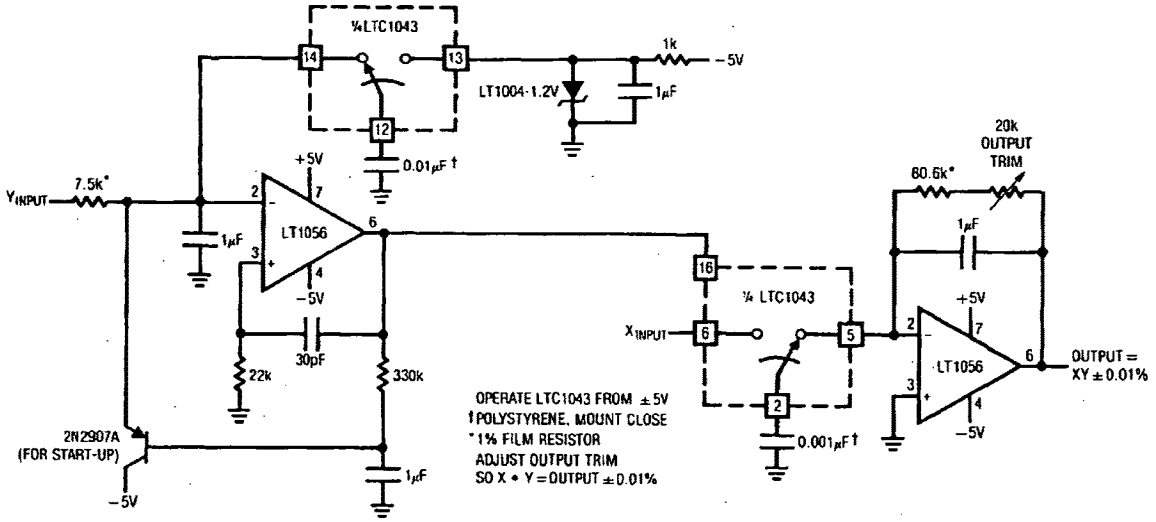


Figure 19. Analog Multiplier with 0.01% Accuracy

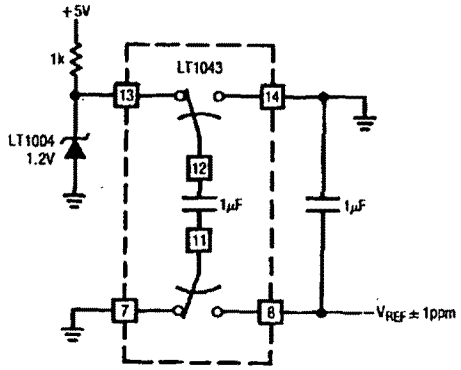


Figure 20. Precision Voltage Inverter

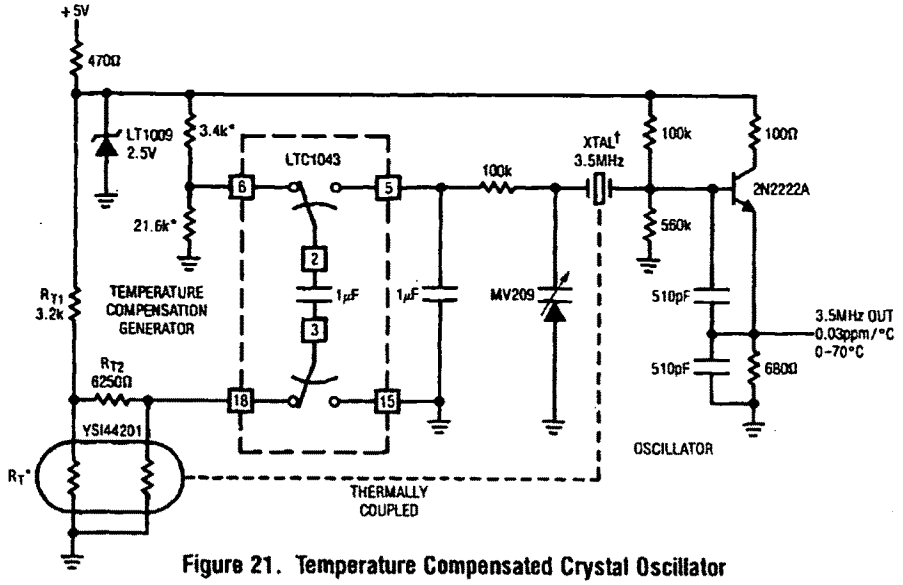


Figure 21. Temperature Compensated Crystal Oscillator

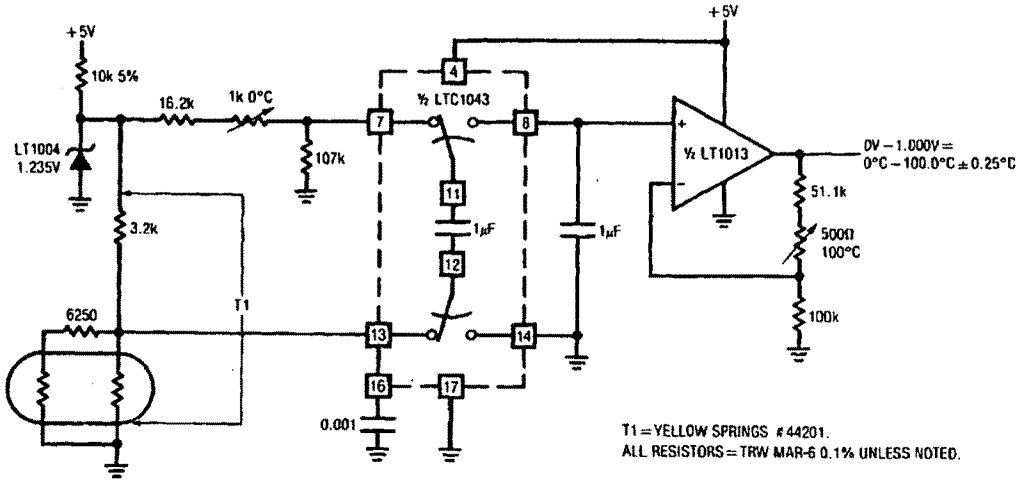


Figure 22. Linear Thermometer

High Current, "Inductorless," Switching Regulator

Figure 23 shows a high efficiency battery driven regulator with a 1A output capacity. Additionally, it does not require an inductor, an unusual feature for a switching regulator operating at this current level.

The LTC1043 switched-capacitor building block provides non-overlapping complementary drive to the Q1–Q4 power MOSFETs. The MOSFETs are arranged so that C1 and C2 are alternately placed in series and then in parallel. During the series phase, the +12V battery's current flows through both capacitors, charging them and furnishing load current. During the parallel phase, both capacitors deliver current to the load. Traces A and B, Figure 24, are the LTC1043-supplied drives to Q3 and Q4, respectively. Q1 and Q2 receive similar drive from pins 3 and 11. The diode-resistor networks provide additional non-overlapping drive characteristics, preventing simultaneous drive to the series-parallel phase switches. Normally, the output would be one-half of the supply voltage, but C1 and its associated components close a feedback loop, forcing the

output to 5V. With the circuit in the series phase, the output (Trace C) heads rapidly positive. When the output exceeds 5V, C1 trips, forcing the LTC1043 oscillator pin (Trace D) high. This truncates the LTC1043's triangle wave oscillator cycle. The circuit is forced into the parallel phase and the output coasts down slowly until the next LTC1043 clock cycle begins. C1's output diode prevents the triangle down-slope from being affected and the 100pF capacitor provides sharp transitions. The loop regulates the output to 5V by feedback controlling the turn-off point of the series phase. The circuit constitutes a large scale switched-capacitor voltage divider which is never allowed to complete a full cycle. The high transient currents are easily handled by the power MOSFETs and overall efficiency is 83%.

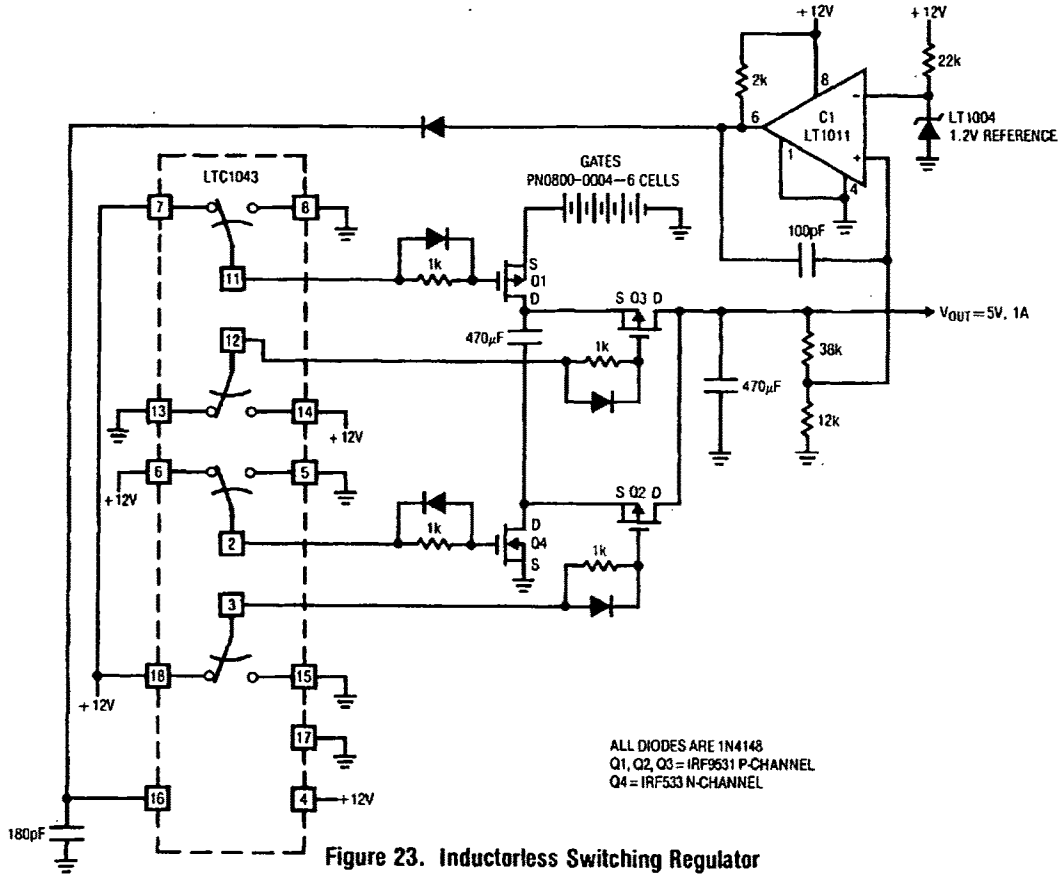


Figure 23. Inductorless Switching Regulator

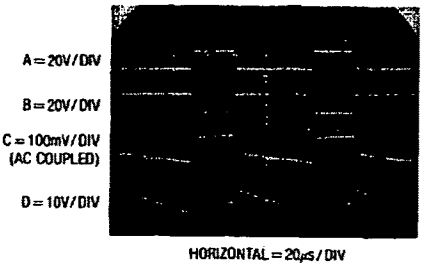


Figure 24

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

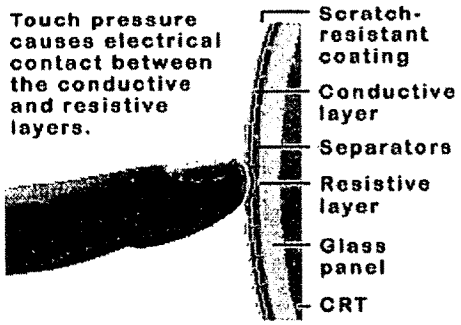
Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☑ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☑ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☑ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

5-Wire Resistive Touchscreens



5-Wire Resistive touch technology consists of a glass or acrylic panel that is coated with electrically condictive and resistive layers. The thin layers are separated by invisible separator dots. When operating, an electrical current moves through the screen. When pressure is applied to the screen the layers are pressed together, causing a change in the electrical current and a touch event to be registered.

5-Wire Resistive type touch screens are generally more durable than the similiar 4-Wire Resistive type. Although clarity is less than with other touch screen types, resistive screens are very durable and can be used in a variety of environments. This type of screen is recommended for demanding point-of-sale systems, restaurant systems, industrial controls, and other workplace applications.

Advantages

- High touch resolution
- Pressure sensitive, works with any stylus
- Not affected by dirt, dust, water, or light
- More durable then 4-Wire Resistive technology

Disadvantages

- 75 % clarity
- Resistive layers can be damaged by a sharp object

Touchscreen Specifications

Touch Type:	Elo AccuTouch 5-Wire Resistive
Cable Interface:	PC Serial/COM Port or USB Port
Touch Resolution:	4096 x 4096
Response Time:	21 ms. at 9600 baud
Light Transmission:	80% +/-5% at 550 nm wavelength (visible light spectrum)
Expected Life:	35 million touches at one point
Temperature:	Operating: -10°C to 50°C Storage: -40°C to 71°C
Humidity:	Operating: 90% RH at max 35°C Storage: 90% RH at max 35°C for 240
Chemical Resistance:	Acetone, Methylene chloride, Methyl ethyl ketone, Isopropyl alcohol, Hexane, Turpentine, Mineral spirits, Unleaded Gasoline, Diesel Fuel, Motor Oil, Transmission Fluid, Antifreeze, Ammonia based glass cleaner, Laundry Detergents, Cleaners (Formula 409, etc.), Vinegar, Coffee, Tea, Grease, Cooking Oil, Salt
Regulations:	UL, CE, TUV, FCC-B

S **/are Drivers:** Windows XP, 2000, NT, ME, 98, 95, 3.1, DOS, Macintosh O_ , _inux, Unix (3rd Party)

A Brief Overview of Gesture Recognition

A primary goal of gesture recognition research is to create a system which can identify specific human gestures and use them to convey information or for device control. To help understand what gestures are, an examination of how other researchers view gestures is useful. How do biologists and sociologists define "gesture"? How is information encoded in gestures? We also explore how humans use gestures to communicate with and command other people. Furthermore, engineering researchers have designed a variety of "gesture" recognition systems - how do they define and use gestures?

Biological and Sociological Definition and Classification of Gestures

From a biological and sociological perspective, gestures are loosely defined, thus, researchers are free to visualize and classify gestures as they see fit. Speech and handwriting recognition research provide methods for designing recognition systems and useful measures for classifying such systems. Gesture recognition systems which are used to control memory and display, devices in a local environment, and devices in a remote environment are examined for the same reason.

People frequently use gestures to communicate. Gestures are used for everything from pointing at a person to get their attention to conveying information about space and temporal characteristics [Kendon 90]. Evidence indicates that gesturing does not simply embellish spoken language, but is part of the language generation process [McNeill 82].

Biologists define "gesture" broadly, stating, "the notion of **gesture** is to embrace all kinds of instances where an individual engages in movements whose communicative intent is paramount, manifest, and openly acknowledged" [Nespoulous 86]. Gestures associated with speech are referred to as *gesticulation*. Gestures which function independently of speech are referred to as *autonomous*. Autonomous gestures can be organized into their own communicative language, such as American Sign Language (ASL). Autonomous gestures can also represent motion commands. In the following subsections, some various ways in which biologists and sociologists define gestures are examined to discover if there are gestures ideal for use in communication and device control.

Gesture Dichotomies

One classification method categorizes gestures using four dichotomies: act-symbol, opacity-transparency, autonomous semiotic (semiotic refers to a general philosophical theory of signs and system that deals with their function in both artificially constructed and natural languages) -multisemiotic, and centrifugal-centripetal (intentional) [Nespoulous 86].

The act-symbol dichotomy refers to the notion that some gestures are pure actions, while others are intended as symbols. For instance, an action gesture occurs when a person chops wood or counts money, while a symbolic gesture occurs when a person makes the "okay" sign or puts their thumb out to hitchhike. Naturally, some action gestures can also be interpreted as symbols (*semiogenesis*), as illustrated in a spy novel, when an agent carrying an object in one hand has important meaning. This dichotomy shows that researchers can use gestures which represent actual motions for use in controlling devices.

The opacity-transparency dichotomy refers to the ease with which others can interpret gestures. Transparency is often associated with *universality*, a belief which states that some gestures have standard cross-cultural meanings. In reality, gesture meanings are very culturally dependent. Within a society, gestures have standard meanings, but no known body motion or gesture has the same meaning in all societies [Birdwhistell 70]. Even in ASL, few signs are so clearly transparent that a non-signer can guess their meaning without additional clues [Klima 74]. Fortunately, this means that gestures used for device control can be freely chosen. Additionally, gestures can be culturally defined to have specific meaning.

The centrifugal-centripetal dichotomy refers to the intentionality of a gesture. Centrifugal gestures are directed toward a specific object, while centripetal

gestures a. ..ot [Sousa-Poza 77]. Researchers usually are concerned with ge..ures which are directed toward the control of a specific object or ... communication with a specific person or group of people.

Gestures which are elements of an autonomous semiotic system are those used in a gesture language, such as ASL. On the other hand, gestures which are created as partial elements of multisemiotic activity are gestures which accompany other languages, such as oral ones [Lebrun 82]. Gesture recognition researchers are usually concerned with gestures which are created as their own independent, semiotic language, though there are some exceptions.

Gesture Typologies

Another standard gesture classification scheme uses three categories: arbitrary, mimetic, and deictic [Nespoulous 86].

In mimetic gestures, motions form an object's main shape or representative feature [Wundt 73]. For instance, a chin sweeping gesture can be used to represent a goat by alluding to its beard. These gestures are intended to be transparent. Mimetic gestures are useful in gesture language representations.

Deictic gestures are used to point at important objects, and each gesture is transparent within its given context. These gestures can be specific, general, or functional. Specific gestures refer to one object. General gestures refer to a class of objects. Functional gestures represent intentions, such as pointing to a chair to ask for permission to sit. Deictic gestures are also useful in gesture language representations.

Arbitrary gestures are those whose interpretation must be learned due to their opacity. Although they are not common in a cultural setting, once learned they can be used and understood without any complimentary verbal information. An example is the set of gestures used for crane operation [Link-Belt 87]. Arbitrary gestures are useful because they can be specifically created for use in device control. These gesture types are already arbitrarily defined and understood without any additional verbal information.

Voice and Handwriting Recognition: Parallel Issues for Gesture Recognition

Speech and handwriting recognition systems are similar to gesture recognition systems, because all of these systems perform *recognition* of something that moves, leaving a "trajectory" in space and time. By exploring the literature of speech and handwriting recognition, classification and identification schemes can be studied which might aid in developing a gesture recognition system.

Typical speech recognition systems match transformed speech against a stored representation. Most systems use some form of spectral representation, such as spectral templates or hidden Markov models (HMM). Speech recognition systems are classified along the following dimensions [Rudnicky 94]:

· Speaker dependent versus Independent: Can the system recognize the speech of many different individuals without training or does it have to be trained for a specific voice? Currently, speaker dependent systems are more accurate, because they do not need to account for large variations in words.
· Discrete or Continuous: Does the speaker need to separate individual words by short silences or can the system recognize continuous sentences? Isolated-word recognition systems have a high accuracy rate, in part because the systems know when each word has ended.
· Vocabulary size: This is usually a task dependent vocabulary. All other things being equal, a small vocabulary is easier to recognize than a large one.
· Recognition Rate: Commercial products strive for at least a 95% recognition rate. Although this rate seems very high, these results occur in laboratory environments. Also, studies have shown that humans have an individual word recognition rate of 99.2% [Pisoni 85].

State of the art speech recognition systems, which have the capability to understand a large vocabulary, use HMMs. HMMs are also used by a number of gesture recognition systems (see Control of Memory and Display). In some speech recognition systems, the states of an HMM represent phonetic units. A

state tran. .n defines the probability of the next state's occurrence. See Fig. 1 for a simple example representation of an HMM. The term *hidden* refers to the type of Markov model in which the observations are a probabilistic function of the current state. A complete specification of a hidden Markov model requires the following information: the state transition probability distribution, the observation symbol probability distribution, and the initial state distribution. An HMM is created for each word (string of phonemes) in a given lexicon. One of the tasks in isolated speech recognition is to measure an observed sequence of phonetic units and determine which HMM was most likely to generate such a sequence [Ljolje 91] [Rabiner 89].

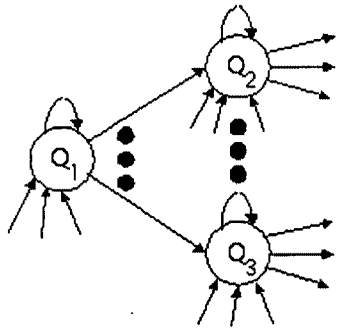


Figure 1: A Simplified Representation of a Hidden Markov Model. The Various Q's Represent the States, While the Lines Represent the Transitions.

From some points of view, handwriting can be considered a type of gesture. On-line (also called "real time" or "dynamic") recognition machines identify handwriting as a user writes. On-line devices have the advantage of capturing the dynamic information of writing, including the number of strokes, the ordering of strokes, and the direction and velocity profile of each stroke. On-line recognition systems are also interactive, allowing users to correct recognition errors, adapt to the system, or see the immediate results of an editing command.

Most on-line tablets capture writing as a sequence of coordinate points. Recognition is complicated in part, because there are many different ways of generating the same character. For example, the letter E's four lines can be drawn in any order.

Handwriting tablets must take into account character blending and merging, which is similar to the continuous speech problem. Also, different characters can look quite similar. To tackle these problems, handwriting tablets pre-process the characters, and then perform some type of shape recognition. Preprocessing typically involves properly spacing the characters and filtering out noise from the tablet. The more complicated processing occurs during character recognition.

Features based on both static and dynamic character information can be used for recognition. Some systems using binary decision trees prune possible characters by examining simple features first, such as searching for the dots above the letters "i" and "j". Other systems create zones which define the directions a pen point can travel (usually eight), and a character is defined in terms of a connected set of zones. A lookup table or a dictionary is used to classify the characters.

Another scheme draws its classification method from signal processing, in which curves from unknown forms are matched against prototype characters. They are matched as functions of time or as Fourier coefficients. To reduce errors, an elastic matching scheme (stretching and bending drawn curves) is used. These methods tend to be computationally intensive.

Alternati\ , pen strokes can be divided into basic components, which are t.. connected by rules and matched to characters. This method is ..ed Analysis-by-Synthesis. Similar systems use dynamic programming methods to match real and modeled strokes.

This examination of handwriting tablets reveals that the dynamic features of characters make on-line recognition possible and, as in speech, it is easier to recognize isolated characters. Most systems lag in recognition by more than a second, and the recognition rates are not very high. They reach reported rates of 95% due only to very careful writing. They are best used for filling out forms which have predefined prototypes and set areas for characters. For a more detailed overview of handwriting tablets, consult [Tappert 90].

Presentation, Recognition, and Exploitation of Gestures in Experimental Systems

The researchers who create experimental systems which use "gestural" input use their own definitions of gesture, which are as diverse as the biological and sociological definitions. The technology used to recognize gestures, and the response derived from gestures, further complicates this issue. Gestures are interpreted to control computer memory and displays or to control actuated mechanisms. Human-computer interaction (HCI) studies usually focus on the computer input/output interface [Card 90], and are useful to examine for the design of gesture language identification systems. Many telerobotic studies analyze the performance of remotely controlled actuated mechanisms. The study of experimental systems which span both HCI and telerobotics will illuminate criteria for the design of a gestural input device for controlling actuated mechanisms located in remote environments.

In such systems, gestures are created by a static hand or body pose, or by a physical motion in two or three dimensions, and can be translated by computer into either symbolic commands or trajectory motion commands. Examples of symbolic command gestures are stop, start, and turn. Gestures may also be interpreted as letters of an alphabet or words of a language. Alternatively, the kinematic and dynamical content of a gesture can represent a trajectory motion command. Combinations of symbolic and trajectory commands are possible in a single gesture.

Given that gestures are used to communicate information, the question arises: is it possible to consistently capture that information in a usable form? Based on the work of Wolf [Wolf 87] in studying hand drawn gestures, we analyze gestural control devices by examining the following generation, representation, recognition, and transformation questions:

· What is the gesture lexicon?
· How are the gestures generated?
· How does the system recognize gestures?
· What are the memory size and computational time requirements for gesture recognition?

And for using gestures for device control:
· What devices are the gestures intended to control?
· What system control commands are represented by gestures?
· How are the gestures transformed into commands for controlling devices?
· How large are the physical and temporal separations between command initiation and response (local control versus telerobotics)?

Based on the answers to the above questions gestural control devices are classified into three categories: control of computer memory and display, local control of actuated mechanisms, and remote control of actuated mechanisms.

Control of Memory and Display

A number systems have been designed to use gestural input devices to control computer memory and display. These systems perceive gestures through a variety of methods and devices. While all the systems presented identify gestures, only some systems transform gestures into appropriate system specific commands. The representative architecture for these systems is show in Figure 2.

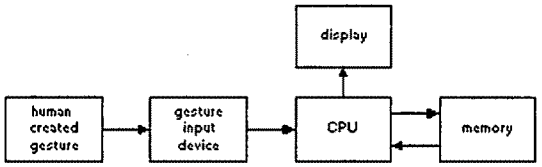


Figure 2: Block Diagram of Architecture for Gestural Control of Memory and Display.

A basic gesture input device is the word processing tablet. Through the use of mouse drawn gestures, Mardia's system allows the editing of display graphics [Mardia 93]. Similarly, Rubine's system uses gestures (stylus markings) on a graphics tablet to represent word processing commands [Rubine 91].

In these representative systems, two dimensional hand gestures are sent via an input device to the computer's memory and appear on the computer monitor. These symbolic gestures are identified as editing commands through geometric modeling techniques. The commands are then executed, modifying the document stored in computer memory. Both systems require that the gesture be completed before recognition can begin.

Both systems require the calculation of feature points to represent the gesture. Mardia's system requires 27 binary valued features. Most features are computed through simple additions and comparisons, but one feature is computed using a Linear Least Squares algorithm. A decision tree is used to sort through the 27 features to determine a specific gesture classification.

Rubine's system requires the computation of 13 different features, with five features requiring multiplications and additions based on the total number of data points used to create the gesture. Again, a linear evaluation function is used to discriminate between gestures.

Murakami's system inputs the gesture data via a data-glove [Murakami 91]. Instead of geometric modeling, a neural network identifies gestures based on a finger alphabet containing 42 symbols. The data-glove, which provides 13 features, allows a wider range of motions than the mouse and stylus input devices, which enables a richer vocabulary of gestural inputs and responses. Several hours are required to train the neural network using a SUN/4 workstation.

Darrell's monocular vision processing system accurately identifies a wide variance of yes/no hand gestures [Darrell 93]. Gestures are represented by a view-based approach, and stored patterns are matched to perceived gestures using dynamic time warping. View-based vision approaches also permit a wide range of gestural inputs when compared to the mouse and stylus input devices. Computation time is based on the number of view-based features used (20 in their example) and the number of view models stored (10). Through the use of specialized hardware and a parallel architecture, processing time is less than 100 ms.

Baudel's system [Baudel 93] uses symbolic gestures to control a Macintosh hypertext program. Gestures are tracked through the use of a data-glove and converted into hypertext program commands. Because Baudel's system identifies natural hand gestures unobtrusively with a data-glove, it is more intuitive to use than a standard mouse or stylus control system. Unlike Murakami's and Darrell's systems which only identify gestures, this system uses gestural commands to control a display program, similar to the control of word processing programs. This system had difficulty identifying gestures which differed in their dynamic phase, for instance, when one gesture was twice the speed of another.

The ALIVE II system [Maes 95] identifies full body gestures, as opposed to hand gestures, through basic image processing techniques. Body gestures are used to control simulated mechanisms ("virtual creatures") located in computer memory ("virtual environment"). ALIVE II is the first system presented here which uses kinematic information from a gesture as part of a control command. The direction of the pointing arm is translated into a virtual creature's travel direction command. The body gesture itself is superimposed into the computer generated environment. A gesture interacts with a virtual environment, and the system can use background environment information to aid in interpreting the gesture command. Therefore, the user can use gestures to point at or grab objects in the virtual environment.

The Digital Desk Calculator [Wellner 91] is another kinematic gesture tracking system which controls a display. Computer graphics are displayed onto a real world desk top. The system tracks a human finger which points at real desk top items. If the system can identify the objects or text pointed at, then the graphical output display responds. Objects pointed to on the graphics overlay also effect the graphics display. For example, a user points at numbers located on a real document. The vision system scans the numbers. When the user points at a virtual calculator, the numbers appear on the calculator and are entered into computer memory. The user can perform operations by pointing at the virtual calculator's buttons superimposed on the desk. Low resolution image processing is used to obtain the approximate position of the finger through motion detection via image differencing. Then, high resolution image processing is used to determine the finger's exact position. Tracking occurs at 7 frames per second.

Starner and Pentland's system uses an HMM method to recognize forty American Sign Language gestures [Starner 95]. Acquired feature vectors are run through all possible sets of five-word sentences. The probability of the data stream being generated by each HMM model is then determined. The system picks the model which has the highest probability of generating the data stream. States consist of a probability distribution of the hand features (as opposed to the pose itself).

Weng's SHOSLIF identifies the most discriminating features of an image through a multi-class, multivariate discriminant analysis [Cui 94]. These features are categorized in a space partition tree. SHOSLIF can recognize 28 ASL signs. The calculations are multiplication intensive, based on the size of the image, the number of images in a gesture sequence, and the number of signs in the lexicon. However, the time spent searching the partition tree increases only logarithmically.

Local Control of Actuated Mechanisms.

An examination of research in mobile robotics and visual servoing systems reveals systems in which control can be conceptualized as local gestural control of actuated mechanisms. Although the "gestures" in many of these systems are created by the local environment rather than a human, interesting possibilities are suggested by analogy. A representative system architecture is shown in Figure 3.

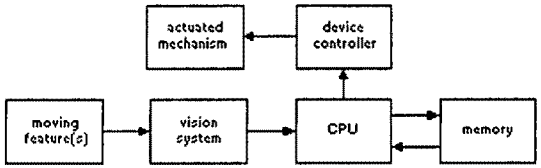


Figure 3: Block Diagram of Architecture for Local Control of Actuated Mechanisms.

Road-following systems used by some mobile robots can be conceptualized as examples of local control through gestures. Two representative systems are Crisman's [Crisman 93] and Dickmanns' [Dickmanns 88] work. Crisman's system uses road features detected through monocular vision as navigation cues

for a mobi. .obot platform. On a reduced color image, classification is perfo. .ed by determining the probability of each pixel representing a r. . surface. Detection of straight roads and intersections on a SUN/4 takes 6 seconds and 20 seconds, respectively. Dickmanns' system models road features, from motion relative to the moving mobile robot, as dynamical systems. Each feature's trajectory is a gesture used to control the attached mobile platform. A Linear Least Squares calculation (with additional computations to handle noise) is used to recognize gestures at video field rate.

The Buhgler Juggler [Rizzi 92] can also be considered a local gesture control device. The system identifies and visually tracks a falling ball "gesture" and uses the dynamics of the gesture to command a robot arm to juggle via a "mirror law" calculation at field rate.

Allen's system [Allen 93] tracks a moving toy train, which can be considered a gesture, to be grasped by a robot arm. Using optical flow, updated centroids of the train are obtained every 0.25 seconds, with a probabilistic method used to obtain a model of the motion.

Kortenkamp uses static gestures, in the form of six human arm poses, [Kortenkamp 96], to control a mobile robot. Small regions in 3-D visual space, proximity spaces, are used to concentrate the system's visual attention on various parts of a human agent. Vision updates occur at frame rate, but the system can only track gestures which move less than 36 deg/sec. This architecture stresses the need to link gestures to desired robot actions. The system can also determine the intersection of a line formed from a pointing gesture with the floor of an environment, allowing a human to give a position command to a mobile robot.

Cohen developed a system which recognizes gestures which are natural for a human to create, consisting of oscillating circles and lines [Cohen 96]. Each gesture is modeled as a linear-in-parameters dynamical system with added geometric constraints to allow for real time gesture recognition using a small amount of processing tie and memory. The linear least squares method is used to determine the parameters which represent each gesture. A gesture recognition and control architecture is developed which takes the position of a feature and determines which parameters in a previously defined set of predictor bins best fits the observed motion. The gesture classification is then used to create a reference trajectory to control an actuated mechanism.

Remote Control of Actuated Mechanisms: Telerobotics.

Telerobotics refers to the control of mechanisms "at a distance". This includes both short and long range control - the mechanism can be in the next room or in earth orbit. Telerobotics also incorporates the study of relevant human-machine interfaces, so as to enable better overall control of mechanisms at a distance.

As above, the control of an actuated mechanism located in the same environment as the input device is "local control in a local environment". Remote control and telerobotics are terms used when the input device and controlled mechanism are located in physically separate environments, requiring visual communication links to enable users to see the activity of the controlled mechanisms.

Certain challenges arise when controlling mechanisms located a very large distance, due to the effect of communication time delays [Sheridan 93]. For example, when relaying a control command from the earth to synchronous earth orbit, the round trip time delay for visual feedback is several tenths of a second. The range of possible controls is limited because of added instabilities arising from time delays in feedback loops.

Research to mitigate the effects of time delays has taken the form of studying system modeling and sensor display concepts [Liu 93]. Conway uses a simulator to aid in mechanism device control, exploiting a time-desynchronized planning model that projects the future trajectory of the telerobotic mechanism (see [Conway 90] and the patent [Conway 91]). Herzinger adds additional sensors to aid user mechanism control [Hirzinger 90].

The time delay problems can also be alleviated if a symbol, representing a recognized gesture command, is sent to the remote site. However, this creates an

additional "one delay": the time required to recognize the gesture. The Robo[illegible]st architecture uses such a gestural control method. The Roboge[illegible]ystem controls a robot with static hand gestures, which are recognized, converted into a command, and sent to a remote site [Schlenzig 94].

A hidden Markov model describes the six hand gestures, and recognition occurs at 0.5 Hz. Each gesture is paired with various robot behaviors, such as turn, accelerate, and stop.

Other telerobotic systems use a joystick to control the remote mechanism (see Figure 4) [Sheridan 93]. Consistent with our previous observations, a joystick is used to input "gestures" which control the remote actuated mechanism.

Unfortunately, if a gesture is generated at a local site and interpreted at a remote site, time delays cause further gesture recognition difficulties: the recognition must now take into account blurred and warped gestures. With additional processing, Darrell's system (see Control of Memory and Display) [Darrell 93] could help identify "warped"gestures.

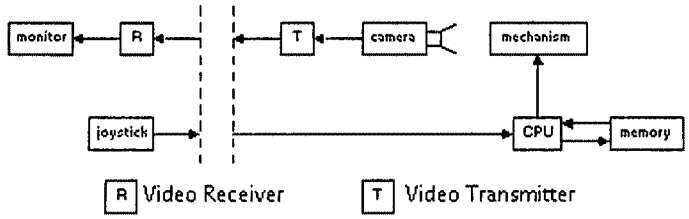


Figure 4: Block Diagram of Longer Distance Telerobotics.

A special communication environment could be developed to take advantage of gestural inputs for the control of devices. Since gestures are visual, a video communication system could transmit a gesture from the local site to the remote site. However, a created gesture would be isolated from the device it is intended to control. If we desire gestures which appear as though they were created in the remote location, another architecture or communication environment is required. (For example, see the University of Michigan M-ROVER remote control project and a patent [Conway 97] on remote control in the visual information stream.)

Issues Concerning the Recognition and Generation of Human Generated Gestures

In addition to human-to-human communication research, the use of gestures as computer input has been studied. Wolf's research in this area examines the following concerns [Wolf 87a]:

1. How consistent are people in their use of gestures?
2. What are the most common gestures used in a given domain, and how easily are they recalled?
3. Do gestures contain identifiable spatial components which correspond to the functional components of command (the action to be performed), scope (the object to which the command is applied), and target (the location where the object is moved, inserted, copied, etc.)?
4. What kind of variability exists in a gesture, and are the deviations predictable or random?

Wolf explored these questions by studying hand drawn two dimensional gestures which were used to edit text documents [Wolf 87a]. Hauptmann attempted to answer similar questions by examining three dimensional spatial gestures which were used to rotate, translate, and scale three dimensional graphic objects

on a comp r screen [Hauptmann 89]. Although the uses were drastically di- ent (text versus graphics), the concepts and conclusions were s...ingly similar.

People consistently used the same gestures for specific commands. In particular, the recognition scheme used by Hauptmann was able to classify the types of three dimensional gestures used for rotation, translation, and scaling with a high degree of consistency.

People are also very adept at learning new arbitrary gestures. Gesturing is natural for humans, and only a short amount of training is required before people can consistently use new gestures to communicate information or control devices (see [Wolf 87], [Hauptmann 89], and [Harwin 90]).

Wolf also discovered that, without prompting, test subjects used very similar gestures for the same operations, and even used the same screen drawn gestures days later. Hauptmann also found a high degree of similarity in the gesture types used by different people to perform the same manipulations. Test subjects were not coached beforehand, indicating that there may be intuitive, common principles in gesture communication. Therefore, this research illustrates that accurate recognition of gestures from an intuitively created arbitrary lexicon is possible.

In both studies, humans were used to interpret the gestures, and humans are very adept at handling noisy data and recognizing shifted data [Shepard 71]. Unfortunately, the problems of computers recognizing specific gestures were not addressed. However, these studies were among the first steps toward designing such a system.

A further complication in gesture recognition is determining which features of the gesture generator are used. In a study of representing language through gestures, McNeill and Levy [McNeill 82] have identified gestures according to physical properties. These properties include hand configuration, orientation of the palm, and direction of movement. McNeill and Levy note that gestures have a preparatory phase, an actual gesture phase, and a retraction phase (see Figure 5). Hauptmann defines a "focus" or "center" of a gesture, used to identify and classify three dimensional gestures. Systems that are created to recognize gestures must define which aspects of the gesture creator they are recognizing.

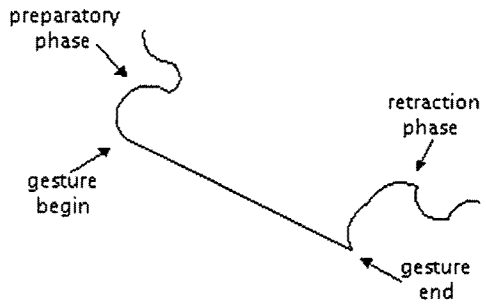


Figure 5: Phases in the Creation of a Gesture.

Wolf noted that screen drawn gesture variability can cause problems when one gesture begins to look like another. However, even when the gesture was not precise, enough information was conveyed to allow the system to execute the correct response. In Hauptmann's study, the variations of three dimensional gestures was not explored as much as the alignment of a gesture with respect to a manipulated object. Still, despite deviations in alignment, there was a high percentage of correct recognition.

Although, m Wolf's and Hauptmann's studies, we know that humans can r. ognize and differentiate between gestures, additional problems a.. involved in designing a system which can reliably recognize human created gestures. There are many difficulties involved because, unlike a device, a human cannot form "perfect" circles, lines, and ASL gesture motions. Furthermore, people often hesitate when starting or ending a gesture, which could confuse a recognition system.

When a system, for example, recognizes a repeated oscillatory gesture, it should not matter where on the circle or line the generation begins. To be more specific, starting a circle gesture at the top or bottom of its arc does not effect how humans recognize the gesture. Therefore, any recognition system should be able to handle such arbitrary initial positions.

Furthermore, how accurately can humans create oscillatory circles and lines of various frequencies? The human arm is a linked kinematic chain, and such oscillating motions might not be easy for it to make. In addition, between the slowest and fastest physically possible speed, gradations in oscillating frequency are possible. However, the gesture velocities for similar spatial gestures need to be distinct in order for any gesture recognition system, human or otherwise, to have a chance at accurate recognition.

Therefore, the gesture velocities for "slow" and "fast" need to separated by a distinct amount. This amount can be determined empirically by watching people make circles and determining when the velocities can be consistently distinguished. Brown [Brown 90] has determined that humans can stay within 5% of the velocity in a desired line motion. Her experiments show that people are very adept at choosing distinct motion speeds when the human pretends there is a visual target velocity to track. This trait is common in all humans and is one of the first aspects to be lost when the human brain is damaged in a specific way.

System Architecture Concepts for Gesture Recognition Systems

Based on the use of gestures by humans (see Biological section), the analysis of speech and handwriting recognizing systems (see Voice and Handwriting Recognition section), and the analysis of other gesture recognition systems (see Experimental Systems section) requirements for a gesture recognition system can be detailed. Some requirements and tasks are:

· Choose gestures which fit a useful environment.
· Create a system which can recognize non-perfect human created gestures.
· Create a system which can use a both a gesture's static and dynamic
· information components.
· Perform gesture recognition with image data presented at field rate (or as fast as possible).
· Recognize the gesture as quickly as possible, even before the full gesture is completed.
· Use a recognition method which requires a small amount of computational time and memory.
· Create an expandable system which can recognize additional types of gestures.
· Pair gestures with appropriate responses (language definitions or device command responses).
· Create an environment which allows the use of gestures for remote control of devices.

Gesture Rocognition Home Page

For a list of current research availible on the internet, please go to the Gesture Recognition Home Page. Also, if anyone wishes to add their page to the GRHP, or knows of pages which should be added, please contact ccohen@cybernet.com.

Bibliography

[Allen 93] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE Transactions on Robotics and Automation*, 9(2):152-165, April 1993.

[Baudel 93] Thomas Baudel and Michel Beaudouin-Lafon. CHARADE: Remote control of objects using free-hand gestures. *Communications of the ACM*, 36(7):28-35, July 1993.

[Birdwhistell 70] R. L. Birdwhistell. *Kinesics and Context; essays on body motion communication*. Philadelphia, PA, 1970.

[Brown 90] S. Brown, H. Hefter, M. Mertens, and H. Freund. Disturbances in human arm movement trajectory due to mild cerebellar dysfunction. *Journal of neurosurgery psychiatry*, 53(4):306-313, April 1990.

[Card 90] Stuart K. Card, Jock D. Mackinlay, and George G. Robinson. The design space of input devices. In *Proceedings of the CHI '90 Conference on Human Factors in Computing Systems*, pages 117-124, April 1990.

[Cohen 96] Charles J. Cohen, Lynn Conway, and Dan Koditschek. "Dynamical System Representation, Generation, and Recognition of Basic Oscillatory Motion Gestures," 2nd International Conference on Automatic Face- and Gesture-Recognition, Killington, Vermont, October 1996.

[Conway 90] L. Conway, R. Volz, and M. Walker. Teleautonomous systems: methods and architectures for intermingling autonomous and telerobotic technology. *Robotics and Automation*, 6(2):146-158, April 1990.

[Conway 91] L. Conway, R. Volz, and M. Walker. Teleautonomous system and method employing time/position synchrony/desynchrony. U.S. Patent 5,046,022, September 1991.

[Conway 97] L. Conway and C. Cohen. Apparatus and method for remote control using a visual information stream. U.S. Patent 5,652,849, July 1997.

[Crisman 93] Jill Crisman and Charles E. Thorpe. SCARF: A color vision system that tracks roads and intersections. *Robotics and Automation*, 9(1):49-58, February 1993.

[Cui 94] Yunato Cui and John J. Weng. SHOSLIF-M: SHOSLIF for motion understanding (phase I for hand sign recognition). Technical Report CPS 94-68, December 1994.

[Darrell 93] Trevor J. Darrell and Alex P. Penland, Space-time gestures. In *IEEE Conference on Vision and Pattern Recognition*, NY, NY, June 1993.

[Dickmanns 88] E. D. Dickmanns and V. Graefe. Dynamic Monocular machine vision. *Machine Vision and Applications*, pages 223-240, 1988.

[Harwin 90] W. S. Harwin and R. D. Jackson. Analysis of intentional head gestures to assist computer access by physically disabled people. *Journal of Biomedical Engineering*, 12:193-198, May 1990.

[Hauptmann 89] Speech and Gestures for Graphic Image Manipulation. In *Computer Human Interaction 1989 Proceedings*, pages 241-245, May, 1989.

[Hirzinger 90] G. Hirzinger, J. Heindl, and K. Landzettel. Predictor and knowledge based telerobotic control concepts. In *Proc. 1989 IEEE Int. Conf. Robotics and Automation*, pages 1768-1777, Scottsdale, AZ, May 14-19 1989.

[Kendon 90] Adam Kendon. *Conducting Interaction: Patterns of behavior in focused encounters*. Cambridge University Press, Cambridge, 1990.

[Klima 74] E. S. Klima and U. Bellugi. Language in another mode. *Language and Brain: Developmental aspects, Neurosciences research program bulletin*, 12(4):539-550, 1974.

[Kortenkamp 96] David Kortenkamp, Eric Huber, and R. Peter Bonasso. Recognizing and interpreting gestures on a mobile robot. *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI '96)*, 1996.

[Lebrun 82] Y. Lebrun. *Neurolinguistic models of language and speech*, pages 1-30. Academic Press, New York, 1982.

[Link-Belt 87] Link-Belt Construction Equipment Company. Operating Safety: Cranes and Excavators, 1987.

[Liu 93] A. Liu, G. Tharp, L. French, S. Lai, and L. Stark. Some of what one needs to know about using head-mounted displays to improve teleoperator performance. *IEEE Transactions on Robotics and Automation*, 9(5):638-649, October 1993.

[Ljolje 91] Andrej Ljolje and Stephen E. Levinson. Development of an acoustic-phonetic Hidden Markov Model for continuous speech recognition. *IEEE Transactions on Signal Processing*, 39(1):29-39, January 1991.

[Maes 95] Pattie Maes, Trevor Darrell, Bruce Blumberg, and Alex Pentland. The Alive System: Full-body interaction with autonomous agents. In *Computer Animation '95 Conference, IEEE Press*, Geneva, Switzerland, April 1995.

[Mardia 93] K. V. Mardia, N. M. Ghali, T. J. Hainsworth, M. Howes, and N. Sheehy. Techniques for online gesture recognition on workstations. *Image and Vision Computing*, 11(5):283-294, June 1993.

[McNeill 82] D. McNeill and E. Levy. *Conceptual Representations in Language Activity and Gestur*e, pages 271-295. John Wiley and Sons Ltd, 1982.

[Murakami 91] Kouichi Murakami and Hitomi Taguchi. Gesture recognition using recurrent neural networks. *Journal of the ACM*, 1(1):237-242, January 1991.

[Nespoulous 86] Jean-Luc Nespoulous, Paul Perron, and Andre Roch Lecours. *The Biological Foundations of Gestures: Motor and Semiotic Aspects*. Lawrence Erlbaum Associates, Hillsdale, MJ, 1986.

[Pisoni 85] D. Pisoni, H. Nusbaum, and B.Greene. Perception of synthetic speech generated by rule. *Proceedings of the IEEE 73*, pages 1665-1676, November 1985.

[Rabiner 89] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), February 1989.

[Rizzi 92] Alfred A. Rizzi, Louis L. Whitcomb, and D. E. Koditschek. Distributed Real-Time Control of a Spatial Robot Juggler. *IEEE Comput...*, 25(5), May 1992.

[Rubine 91] Dean Rubine. Specifying gestures by example. *Computer Graphics*, 25(4):329-337, July 1991.

[Rudnicky 94] Alexander I. Rudnicky, Alexander G. Hauptmann, and Kai-Fu Lee. Survey of current speech technology. *Communications of the ACM*, 37 (3):52-57, March 1994.

[Schlenzig 94] J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputs using Hidden Markov Models. *Proceedings of the Second Annual Conference on Applications of Computer Vision*, December 1994.

[Shepard 71] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171:701-703, 1971.

[Sheridan 93] T. B. Sheridan. Space teleoperation through time delay: review and prognosis. *IEEE Transactions on Robotics and Automation*, 9(5):592-606, October 1993.

[Sousa-Poza 77] J. F. Sousa-Poza and R. Rohrberg, Body movement in relation to type of information (person- and non-person oriented) and cognitive style (file dependence). *Human Communication Research*, 4(1), 1977.

[Starner 95] Thad Starner and Alex Pentland. Visual recognition of American Sigh Language using Hidden Markov Models. *IEEE International Symposium on Computer Vision*, November 1995.

[Tappert 90] C. Tappert,, C. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Transactions on Patt*

A Usable Real-Time 3D Hand Tracker

Subutai Ahmad
Interval Research Corporation,
1801-C Page Mill Rd., Palo Alto, CA 94304
ahmad@interval.com

Abstract

This paper presents a computer vision system for tracking human hands. The algorithms used to extract the 3D position and planar orientation of the hand, and the joint angles of the fingers are described. The combined system is able to track a natural hand at 30 frames per second on a standard workstation with no special image processing hardware other than a frame grabber. The tracker has been used as an interface for navigating around virtual worlds.

1 Introduction

Is it feasible to build a practical and non-intrusive hand tracking device? The answer is at least a partial "yes". In this paper I give an overview of two working systems, based on computer vision, for extracting the 3D position, planar orientation, and the finger joint angles of natural hands.[1]

The primary motivation of this work has been the development of an ideal interface for manipulating three dimensional objects and for navigating around three dimensional environments. It is clear that our hands are our primary means of interacting with our physical environment. As such our hands have evolved to be versatile but highly complex devices. In fact, as children, we spend years mastering their use. Rather than invent completely new interfaces it is natural to take advantage of this intense training and attempt to exploit the way we use our hands.

There are a number of stringent requirements that must be met in order for a hand tracking system to be really useful as an interface. First, real time performance is critical. In our experience we find that the latency must be smaller than 100 milliseconds for the system to be bearable, and should really be smaller than 50 msecs for extended use. This agrees qualitatively with the psychophysical literature on "the psychological moment"[6] (briefly: visual events that occur within 100 milliseconds are naturally integrated, events that occur outside that time window are not). Second, we must use technologies that keep the user as unencumbered as possible. The system should be able to extract the desired information without requiring the user to wear gloves, wires, or other encumberances as with the DataGlove[9].

[1] Gesture recognition, or the task of interpreting the hand configuration, is beyond the scope of this paper. See [5] for a good introduction to this topic.

These requirements pose challenging problems for computer vision. No general purpose solutions are known. The rest of this paper describes some attempts at solving the task outlined above. The approach is quite specific to hand tracking but will hopefully lead to insights that are more generally applicable.

2 A Four Degree of Freedom Hand Tracker

We now describe the first version of our system, implemented in 1992 at Siemens Central Research, Munich. This system is able to reliably track a normal (unmarked and unencumbered) hand in real time. Unlike [4, 5] the system is relatively insensitive to image clutter and extracts three dimensional positional information. This version of the tracker can run on a standard Sun Sparcstation 10 at a speed of 30 frames per second, without any special image processing hardware other than a framegrabber. (A previous version of the system required a person to wear a specially marked cotton glove[8].) There are essentially two parts to the system: a segmentation module and a control strategy. These are described in turn below.

2.1 Color Histogram Based Segmentation

Given sequences of camera images, an important task of the system is to separate the target hand from the background. The main challenge is to build a segmentation module that is insensitive to normal backgrounds and that operates in real-time. This is not an easy task as the image may contain many irrelevant and confusing details (See Figure 1).

Our algorithm works by estimating the distribution of skin colors on a person's hand. This information is then used to detect which parts of the image belong to the hand and which are part of the background. The hope is that as long as the objects in the scene do not have the same color distribution as the user's hand the segmentation will be relatively clean.

2.1.1 Constructing the Histogram

A color-histogram is used to estimate the distribution of colors in a patch of skin. The basic idea behind a color histogram is to partition RGB color space into a number of bins. To train the system we select a patch of skin from an image (by a "patch" we mean a small square region in the image). From the set of

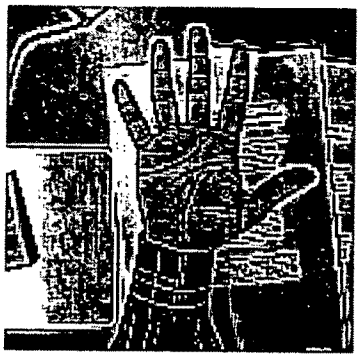


Figure 1: Example of a typical image with a hand. Note the complex background which the system has to ignore.

pixels in this patch, a color histogram is constructed by counting the number of pixels that fall into each bin. By computing a histogram of sample patches from the skin in this way, the system essentially constructs a rough estimate of the probability distribution of colors in the skin. Instead of RGB space we use a normalized 2D space that tends to eliminate the effects of varying illumination. Given an example skin-colored pixel, with color values r, g, and b, we compute its normalized colors as $r' = r/(r+g+b+1)$ and $b' = b/(r+g+b+1)$. These give values between 0 and 1 and can be thought of as computing the percentage of red and percentage of blue. Each of the dimensions r' and b' are discretized by a factor of d. The histogram is then a d x d array.

2.1.2 Histogram Based Segmentation

At run time the tracking system repeatedly matches small patches of the image to the stored histogram using a matching algorithm. To match two histograms, the histogram intersection algorithm [7] is used. The match score $\mathcal{M}$ between histograms p and q is defined as:

$$\mathcal{M}_{p,q} = \frac{\sum_{i,j} \min(H^p(i,j), H^q(i,j))}{\sum_{i,j} H^p(i,j)}. \qquad (1)$$

Those patches whose score is above a threshold (typically 0.9) are skin-colored and assumed to belong to the hand.

Once each image is segmented, a list of patches in the image which match the stored histogram is created. Each patch has associated with it a center and an area. Figure 2 shows an example segmentation given the image in Figure 1. Each small square represents one patch. As can be seen, the color-based segmentation results in a clean separation of the hand from the background.

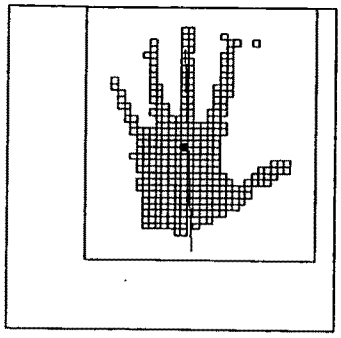


Figure 2: The result of segmenting a hand with a color histogram. Each small unfilled rectangle represents one segmented patch. The filled square represents the center of mass, the line represents the hand's orientation, and the inner large rectangle the search region for the next frame.

2.1.3 Extracting 3D Position

After the segmentation process, it is possible to extract the 3D location and 2D orientation of the hand. The center of mass of the segmented patches is used as the 2D position of the hand: $C_x = \sum_i p_{ix}/N$ and $C_y = \sum_i p_{iy}/N$ where N is the total number of patches that were considered part of the hand, and p_{ix} and p_{iy} denote the x and y coordinate of the center of the i'th patch.

The planar rotation (rotation about the camera axis) is estimated by fitting an ellipse to the segmented patches and computing the angle of its principal axis. This can be done by computing the second order moments of the patches:

$$m_{20} = \sum (C_x - p_{ix})^2 \qquad (2)$$

$$m_{02} = \sum (C_y - p_{iy})^2 \qquad (3)$$

$$m_{11} = \sum (C_y - p_{iy})(C_x - p_{ix}) \qquad (4)$$

Then the orientation of the principal axis is given by:

$$\theta = \frac{1}{2} \tan^{-1}\left(\frac{m_{20} - m_{02}}{2m_{11}}\right) \qquad (5)$$

To smooth out noise we compute a moving average for each of the above parameters. The small filled square and the oblique line in Figure 2 depict the computed center of mass and orientation, respectively, of the hand in Figure 1.

Since we are only using one camera, it is difficult to obtain accurate estimates of the absolute distance of the hand to the camera. However, as an interface, it is sufficient to compute the relative depth (i.e. the depth relative to some standard location). With this information it is possible to tell whether the target is coming closer or moving further away. One way

to compute this is to simply count the total area of the segmented patches. As the target moves closer it will occupy a larger section of the image and the total patch area will increase. However this is susceptable to interference from transient noise patches. We obtain more robust estimates by weighting the area of the patches by a Gaussian placed on the center of mass:

$$A = A_P \sum_i \exp\left(\frac{-(C_x - p_{ix})^2 - (C_y - p_{iy})^2}{S}\right) \quad (6)$$

Here A_P is the area that each patch is responsible for and S is a scaling constant.

Computing relative distance in this way requires a simple calibration step. On the first image the hand is assumed to be at some canonical position. The initial area, A_0, is computed for the initial frame and stored. Then on subsequent frames, depth estimates are obtained by computing the instantaneous area and comparing it with A_0: $C_z = \frac{A_0}{A}$. This quantity will be 1 if the target is exactly at the same position as in the calibration frame. The value will get smaller as the target approaches the camera, and larger as it moves away. Thus it provides an estimate of the relative distance of the target from the camera.

2.2 Control Strategies

Although the above computations can be carried out reasonably efficiently, in order to obtain real-time performance on our system (a Sun Sparcstation), two resource allocation techniques were used. These include a dynamic search window, and a technique for adaptive subsampling. Without these techniques the tracking algorithm can achieve a maximum frame rate of about $8 - 10$ frames/second. The next few sections describe these aspects of the system.

2.2.1 Modifying the Search Region

A dynamic search region is implemented to ignore irrelevant regions of the image. The system maintains a rectangular tracking window around segmented patches. For each subsequent frame only image patches within this window are searched. At each iteration, given the current segmentation, the boundaries for the search region for the next frame are computed as follows:

$x_{min} = \min_i(p_{ix}) - b$ and $x_{max} = \max_i(p_{ix}) - b$
$y_{min} = \min_i(p_{iy}) - b$ and $y_{max} = \max_i(p_{iy}) - b$

Thus at the next frame, only the image pixels in the rectangle defined by (x_{min}, y_{min}) and (x_{max}, y_{max}) are searched. The constant b (we use a value of 40 pixels) ensures that the window is slightly larger than the actual target boundaries.

2.2.2 Adaptive Subsampling

It is not necessary to check every pixel within the search window. Indeed it is too time consuming to do so. In order to speed things up, the system subsamples the image. That is, once an image patch at location (x, y) is checked, the next patch is chosen starting at location $(x + s, y)$. (If the end of the scan line has been reached, then the patch starting at $(0, y + s)$ is checked.) The issue of determining the best subsampling constant, s, is important but non-trivial. Subsampling affects both the accuracy and the speed of the tracking. If s is too large than the position estimates discussed in the last section will be too noisy. If s is too small then too much time will be taken up processing each image.

Rather than use predetermined s, the system uses an adaptive technique to automatically select the best subsampling. The key is to allow the client application which is using the tracker to specify a goal frame rate. The segmentation module then continuously monitors its speed. If the segmentation time is faster than desired, then s is decreased. Conversely, if the time is slower than desired then s is increased to give greater speed.

2.2.3 Balancing Accuracy and Speed at Varying Depths

The above two techniques also solve a common problem concerned with the three-way interaction between accuracy, speed, and depth. In particular, when the target is close to the camera, the resulting image of the hand is large. In this case, the search window will be large, requiring more processing time per frame. s will be automatically increased, maintaining system throughput. When the target is far from the camera, the resulting image is small and so a smaller value of s is required in order to maintain accuracy. In this case the search window will be correspondingly small, requiring less processing per frame, and so the subsampling will be automatically decreased. The end result is that the system maintains relatively constant accuracy and speed at different depths. (It is easy to see that no fixed value of s can achieve this result.)

2.3 Using the Hand Tracker as a User Interface

The tracking system has been used to successfully navigate in virtual environments. Figure 3 shows the setup. The camera images the hand from above. Hand movements are transmitted to a real time VR simulator so that by moving his or her hand the user is able to translate in all six directions (up, down, left, right, forward, and backward) and rotate the view clockwise or counterclockwise. A 3D rendering of the hand provides positional feedback to the user for navigation. By including objects in the world that perform actions when touched the user is able to actually manipulate both the internal and external world. For example, we have implemented a CD player control panel, which when touched, plays back music using a CD ROM.

3 A 19 Degree of Freedom Hand Tracker

So far we have described a system that is able to track the 3D position and planar orientation of the hand. It is also of interest to recover the complete configuration of the hand. From a computer vision point of view, this is a very difficult task as the hand

Figure 3: The view of the hand tracking setup.

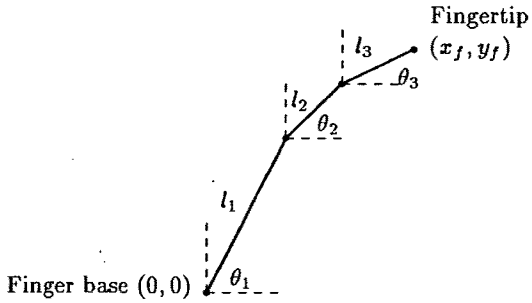


Figure 4: A model of the joint angles in a finger.

contains about 20 degrees of freedom (each finger contains four degrees of freedom: two joints with one degree of freedom and one joint, the base, with two). This section describes one approach for recovering the required information from the image.

3.1 Recovering Joint Angles

Figure 4 shows a schematized side view of a finger and its joints (the fourth degree of freedom, moving a finger from side to side is not considered here). Given the joint angles θ_i and finger segment lengths l_i, we can easily compute the location of the fingertip:

$$x_f = l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3) \quad (7)$$

$$y_f = l_1 \sin(\theta_1) + l_2 \sin(\theta_2) + l_3 \sin(\theta_3) \quad (8)$$

This is complicated by the fact that not every point in joint space is realizable with our fingers. For example, although there are three joints shown, the finger can only cover a two dimensional space. Even in 2D space, not every point is reachable (e.g. we cannot bend our finger backward). It turns out that if location of the fingertip is known, the set of possible joint angles is actually very constrained. This implies that, given the location of the fingertips, it is theoretically possible to approximately compute the actual joint angles. Unfortunately it is difficult to do this directly: there does not seem to be a closed form analytic expression for the inverses to equations (7) and (8) that can easily incorporate the joint angle constraints.

The approach we have taken has been to learn the inverse mapping. The idea is that the forward model is easy to compute. Given the 3 angles and knowledge of the segment lengths, we can compute the location of the fingertip from equations (7) and (8). So it is possible to construct a training set with pairs of vectors consisting of joint angles and corresponding fingertip locations. Using this training set, we can simply train a function approximator to compute the inverse map.

This turns out to be quite successful. For our purposes the best results were obtained using nearest neighbor approximation. That is given a fingertip location, locate the closest fingertip in the training set and select the corresponding joint angles. Using kd-trees one can implement nearest neighbor very efficiently [2]. Average retrieval time is $O(\log n)$ where n is the number of vectors in the training set.

3.2 Fingertip Detection

The joint angle recovery has been incorporated within the hand tracking system described earlier. In order to do this a "fingertip detector" had to be implemented. For efficiency reasons we decided on fitting a very coarse model of the hand to the segmented image patches. The palm is represented as a circle; the fingers are represented by lines emanating from the center of the wrist. The lines are allowed to rotate around the wrist center. To fit such a model we need to locate the radius of the palm and determine the angles and lengths of each finger.

In order to locate the circle representing the palm, the current system assumes that the center of the palm is at the center of mass. This is not completely accurate but is a reasonable assumption as long as the arm does not also appear in the segmented patches (i.e. the person should wear a full sleeved shirt). In order to find the palm the system computes the largest circle such that the area of all patches within the circle is roughly the same as the actual area of the circle.

Once the palm is located, the fingertips need to be detected. Every patch that is outside the circle could be part of a finger. In a human hand the finger bones emanate from a central point on the wrist. Thus the angles of the fingers (especially the thumb) are best measured from the wrist center. This point is obtained by projecting the center of mass backwards along the orientation of the hand to the edge of the circle.

In order to locate the actual angle of each finger we have used a Hough transform based approach. The set of possible angles are discretized into a number of bins. Each patch outside the circle has associated with it an angle to the wrist center and votes for this angle. Within each bin the system also keeps track of the patch with maximal distance that had voted for it. After the voting process the five best peaks in the histogram are selected and the maximal patch is chosen as the fingertip location.

3.3 Real Time Finger Modeling

The fingertip detector and joint angle recovery modules have been combined to obtain a real time system for approximating the user's finger movements. The system works, as before, by using the first frame as a calibration frame. In this frame the users hand is assumed to be flat with all fingers extended. The depth of the hand and the maximal length of each finger are stored. Then on each subsequent frame, each fingertip is located. Treating each finger independently, the system uses the joint angle module to output the joint angles for each finger to a visualizer. Each finger is only allowed three degrees of freedom (fingers are not allowed to wiggle from side to side). This results in 15 degrees of freedom for the fingers. Combined with the position and angle estimates leads to a total of 19 degrees of freedom. The combined system can run at a rate of 10 frames per second on a SparcStation 10. Although it often works well, this system unfortunately is not as stable as the hand tracker. This is due in part to failures in detecting the fingertip when the user's hand is not approximately parallel to the camera plane. When the fingertips are detected correctly then the visualizer shows a reasonable interpretation of the user's fingers moving around. Further research needs to be conducted on the feature detection stage.

4 Discussion

This paper has described initial steps towards building a complete hand recognition system based on computer vision. There are still several drawbacks of the systems described above. First, only rotations parallel to the camera plane are recovered. Rotating the hand around the other two axes can confuse the system. Second, although the system is quite insensitive to the image background, only one skin colored object may be present in the image at any one time. This means that the system cannot yet handle two hands in the image. Finally, the system for recovering joint angles occasionally has problems detecting the fingertip, mainly due to the limitations of the hand model used.

Apart from these disadvantages, the systems are quite stable and robust, particularly the position and orientation tracking component. There is typically very little noise in the position estimates. The approach based on fitting a set of patches to a model has proven to be very robust to noise in the segmentation process. Finally, the speed of the system provides the user with rapid feedback, an important factor in interactive systems.

How far away are we from full scale $3D$ hand tracking? There are a number of outstanding problems that must be solved. There are still no general algorithms for dealing with self-occlusion, for modeling non-rigid objects, and for determining the pose of non-rigid objects. Adaptive techniques based on density estimation[1] and on estimating the surface that data points lie on[3] appear promising. Although the task is formidable, the results outlined here provide some hope that vision based algorithms can eventually lead to non-obstrusive and natural $3D$ user interfaces.

Acknowledgements

Portions of this research were done at Siemens Central Research, Munich, Germany. I thank Daniel Goryn, Christoph Maggioni, Rolf Schuster, Brigitte Wirtz and Volker Tresp for helpful comments and suggestions.

References

[1] S. Ahmad and V. Tresp. Some solutions to the missing feature problem in vision. In S.J. Hanson, J.D. Cowan, and C.L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 393–400. Morgan Kaufmann Publishers, 1993.

[2] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[3] Christoph Bregler and Stephen M. Omohundro. Surface learning with applications to lipreading. In J.D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 43–50. Morgan Kaufmann Publishers, 1994.

[4] M.W. Krueger. *Artificial Reality II.* Addison-Wesley, 1991.

[5] J. Segen. Model learning and recognition of non-rigid objects. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, 1989.

[6] J.M. Stroud. The fine structure of psychological time. In H. Quastler, editor, *Information theory in psychology.* Free Press, Glencoe, Ill., 1956.

[7] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

[8] B. Wirtz and C. Maggioni. Imageglove: A novel way to control virtual environments. In *Proceedings of Virtual Reality Systems '93*, New York, 1993.

[9] T.G. Zimmermann, J. Lanier, C. Blanchard, and S. Bryson. A hand gesture interface device. In *Proceedings ACM CHI+GI Conference: Human Factors in Computing Systems and Graphics Interface*, pages 189–192, 1987.

SWITCHED CAPACITOR INTERFACE CIRCUIT for CAPACITIVE TRANSDUCERS

G. J. YEH, I. OHDO*, W. H. KO

Case Western Reserve University, Cleveland, OHIO, USA
* Tohoku Institute of Technology, Sendai, Japan

ABSTRACT

A capacitance measurement circuit is reported here. The unique features are; (1) The sensitivity and null point of the output can be programed independently by two control voltage sources. (2) The insensitivity of the circuit to large fixed stray capacitance, makes it useful in silicon to silicon capacitive transducers and the two chips module approach of sensor packaging. (3) A high output level, with a sensitivity of 1V/pF is obtained from a discrete component bench model. The principle and operation of this circuit is presented.

INTRODUCTION

Capacitive sensors have been used to design transducers measuring pressure, acceleration, force and position. Interface circuits used are generally either an oscillator with variable frequency [1] or a ratio detector such as a diode bridge [2] to detect the change in capacitance from a null point. The first method does not provide sufficient baseline stability. The second method shows sufficient sensitivity when the percentage of change of capacitance is large. However, in cases such as silicon to silicon capacitor and two chip module assembly, where the stray capacitance may be larger than the capacitance of the sensing element, the sensitivity is greatly reduced. Furthermore variations in stray capacitance will vary the null point of an individual chip, thus requiring some kind of on-chip trimming. Another method using a switched capacitor circuit has been reported [3].

This article presents an interface circuit the output of which is proportional to the change of capacitance. The sensitivity is not affected by the magnitude of parasitic capacitance. Also an external adjustment of the control voltages is provided to set the null point and sensitivity independently at any desired values. This makes it possible to measure capacitance change over several ranges by the zero supresion technique. The sensitivity and output level obtainable are higher than other types of circuits.

In the early stage of miniaturized capacitive pressure transducer development; diode, resistor and capacitor circuits are widely used [4]. These circuits can be divided into two catagories; differential type and ratio type. The differential type circuit detects the variation of the capacitance, while the ratio type circuit senses the ratio of the capacitance variation to the total capacitance. All these circuits convert the charges stored in the sensing capacitor into either a current or voltage output. Basically they are switched capacitor circuits, however, the diodes act as passive switching elements. The diodes turn on and off according to the voltage across them. The MOS switch can be turned on or off independent of the voltage across it. The MOS switches are adapted in the recently developed capacitance measurement circuits. The typical examples are the ratio type CMOS bridge circuit [2] and the differential type switched capacitor read out amplifier [3].

When the manufactured capacitive sensing elements are not uniform enough, trimming is required to have the transducers interchangable. The most straight forward solution is to make the circuit electrically programable. In order to design an electrically programable capacitance measurement circuit, the switched capacitor technique is the most appropriate [5]. However, the circuit should be simple enough such that it can be integrated in a small chip area. The high output impedance of a switched capacitor

CH2127-9/85/0000-0060 $1.00 © 1985 IEEE

simulated resistor depends on the inverse of the normally low capacitance value. Feedback technique can be used to decrease the output impedance.

BLOCK DIAGRAM

The system block diagram is shown in Fig.1.

The building blocks of the circuit are:

VCFCI	voltage, capacitance and frequency controlled current source
R	Current to Voltage converter

And:

$$V_f = V_{out}$$

$$= (I_x - I_o - I_f) R$$

$$= (V_g \cdot C_x - V_o \cdot C_o - V_f \cdot C_f) \cdot f \cdot R \quad (1)$$

$$V_{out} = \frac{(V_g \cdot C_x - V_o \cdot C_o) \cdot f \cdot R}{(1 + C_f \cdot f \cdot R)} \quad (2)$$

when $C_f \cdot f \cdot R \gg 1$,

$$V_{out} = (V_g \cdot C_x - V_o \cdot C_o) / C_f$$

$$S_{Cx} = \partial V_{out} / \partial C_x = V_g / C_f \quad (3)$$

The sensitivity S_{Cx} can be adjusted by varying V_g and the output voltage can be nulled by varying V_o. Notice when $(C_f \cdot f \cdot R)$ is much greater than 1, the output voltage is independent of clock frequency (f) and the value of the current to voltage converter R.

CIRCUIT ANALYSIS

The VCFCI can be realized by using two switches clocked by a two phase nonoverlaping control signal as shown in Fig.2a. The capacitor C is charged to potential V at the end of phase 1, and is discharged through the ammeter A in phase 2. The average current flow I through the ammeter is "$(V \cdot C)/T$" or "$V \cdot C \cdot f$". If the ammeter is replaced by a MOSFET current mirror, and the switches are replaced by the CMOS analog switches, as shown in Fig.2b, the drain current of T_2 will be $I_d =$ "$(V - V_t) \cdot C \cdot f \cdot k$". Where k is the current mirror ratio and V_t is the threshold voltage of MOSFET.

In order to perform the current substraction function in the two phase system, a current substractor "6" as shown in Fig.3, is used. The current through MOSFET T_1, T_2, T_3 and T_4 are I_{d1}, I_{d2}, I_{d3} and I_{d4}.

$$I_{d1} = I_{d2} \quad (4)$$

$$I_{d4} = I_{d3} = I_x - I_o \quad (5)$$

A dummy VCFCI discharging through T_5 is used to cancel the stray capacitance in parallel with C_f. Part of the stray capacitance is from drain and source junction capacitance which is sensitive to the applied voltage across the junction. This voltage dependence effect of the stray p-n junction capacitors will affect the output linearity. Large filter capacitors , C_f, with large capacitance are necessary in the two phase system to average out the pulsed current. This is not desirable in applications where chip size is limitted.

The circuit diagram of the three phase system is shown in Fig.4. Switch 1 and 2 are closed during phase 1. The voltage across C_x is V_g and across C_o is V_o. During phase 2, switch 3 and switch 4 are closed. The charges stored in C_o are discharged through T_1 and the charges in C_x are discharged through T_2. Due to the current mirror T_1 and T_2, the same amount of charge will be drawn from C_x. At the end of phase 2, a charge of $(C_x \cdot V_g - C_o \cdot V_o)$ is left on C_x. During phase 3, the charge left on C_x is discharged through T_3. By measuring the current through T_4, one can determine the amount of charges left on C_x. If the current mirror ratio k is 1, the overall result is,

$$I_4 = (C_x \cdot V_g - V_o \cdot C_o) / T \quad (6)$$

where T is the period of the clock, and

$$\partial I_4 / \partial C_x = V_g \cdot f \quad (7)$$

A dumy VCFCI is used to improve the linearity of the voltage to current feedback element. The two current components $(I_x - I_o)$ and I_f are compared in current mirror T_{10} and T_{11}. Their difference is converted to a voltage across the gate and source of T_9. The VCFCI of I_f is used as the load on T_9 to generate V_{out}.

EXPERIMENTAL RESULT

A sensitivity (S_{Cx}) of 0.5 to 1.0 V/pF is observed. The sensitivity versus control voltage V_g is shown in figure 5. The sensitivity varies less than 1% when the output voltage varies from 6V to 9.4V. The sensitivity versus

C_x is shown in figure 6. In the measurement, the null point of the output is kept at 6V by varying $C_o V_o$. The temperature dependence of the circuit is less than 0.01 pF/°C when C_x is 47 pF. The frequency dependence of the output null point is less than 0.001 pF/KHz in the oscillator frequency range of 200KHz to 240 KHz. Sensitivity varies less than 1% in the oscillator frequency range of 160KHz to 220KHz. Low frequency noise limits the resolution to 0.01pF atroomtemperature.

CONCLUSION

The discrete component bench model shows that this switched capacitor capacitance measurement circuit is programble, insensitive to fixed stray capacitence and has high output level. The low frequency noise is the limitation on the resolution. This problem can be solved by using small MOS switches and large gate MOS transistors for the current mirrors. The higher the RC time constant of the switch resistance and the associated capacitor, the lower the low frequency noise amplitude [7]. The junction capacitance in parallel with C_f should be made equal to the junction capacitance of the dumy VCFC1. This will eliminate the voltage dependent part of the feedback loop.

When higher performance is required, modulation of the capacitance variation signal can be used to improve the DC stability and avoid the low frequency noise. This can be done by using AC control voltage sources instead of the DC voltage sources used in the bench model.

REFERENCE

(1) Sander, C. S., Knutti, J. W. and Meindl, J. D. " A Monolithic Capacitive Pressure Sensor with Pulse-Period Output ", Trans. Electron Devices, ED-27 (5): 927-930 (May, 1980).

(2) Ko, W. H. et al., " A Design of Capacitive Pressure Transducer ". Proc. of IEEE Engineering in Medicine and Biology Society Annual Conference, Los Angeles, California (September 15-17 1984). p.32.

(3) Wise, K. D. " Circuit Techniques for Integrated Solid-State Sensors ". IEDM: 380-383 (1983).

(4) Kerwin W. J. and Schaffer G. L. " A Two-Wire IC Compatible Capacitive Transducer Circuit ", Review of Scientific Instruments, 41 (12): 1783-1788 (December, 1970).

(5) Fudim, E. V.; " Fundamental of the Switched Capacitor Approach to Circuit Synthesis ". IEEE Circuit and System Magazine, Vol.6, No.4, P 12-21.

(6) Degrauwe, M. et al., " Adaptive Biasing CMOS Amplifiers ". IEEE Journal of Solid State Circuits, SC-17 (3): 522-528 (June,1982).

(7) Furrer R. and Guggenbuhl W. " Noise Analysis of Sampled-Data Circuit " Proc. ISCAS, Chicago 1981, p860-863.

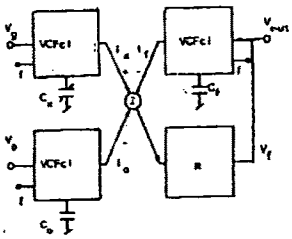
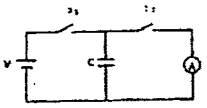


FIGURE 1.
Block Diagram



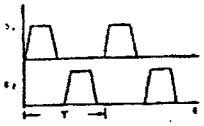


FIGURE 3.a

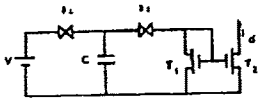


FIGURE 3.b
Voltage Capacitance and Frequency Controlled Current Source

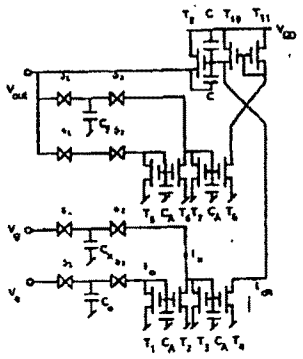


FIGURE 1.
Two Phase System

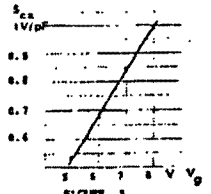


FIGURE 5.
Sensitivity S_{cx} versus
Control Voltage V_g

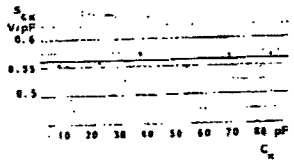


FIGURE 6.
Sensitivity S_{cx} versus
C_x (null point is kept constant)

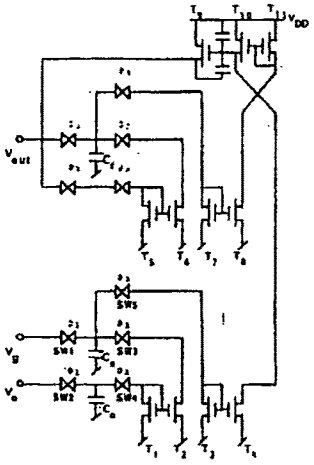
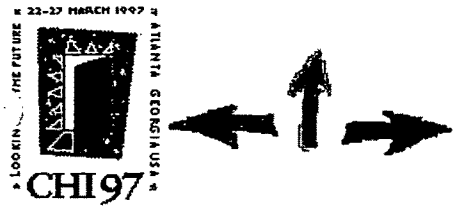


FIGURE 4.
Three Phase System

CHI 97 Electronic Publications: Late-Breaking/Short Talks



Dual Stream Input for Pointing and Scrolling

Shumin Zhai Barton A. Smith Ted Selker

IBM Almaden Research Center,
650 Harry Road, San Jose, California 95120, USA

+1 408 927-1112, {zhai, basmith, selker}@almaden.ibm.com

ABSTRACT

To find ways to improve users' performance of tasks that involve both scrolling and pointing, we studied three dual-stream input methods, with one stream for pointing and one for scrolling. The results showed that a mouse augmented with a tracking wheel did not outperform the conventional single stream mouse. Two other methods, a mouse with an isometric rate-control joystick and a two handed system significantly improved users' performance.

eywords

Input devices, scrolling, dual-stream input, two-handed Input.

INTRODUCTION

Typical GUI interfaces feature a single stream of input for all interaction tasks. Alternative methods, such as two-handed input, have been proposed and demonstrated to be effective in many tasks [e.g. 1, 2]. These ideas are just beginning to be implemented in mainstream user interfaces. This study investigates user's performance and preferences with three new dual stream input methods, in a web browsing task.

During browsing, one often needs to alternatively perform scrolling and pointing. With a traditional scroll bar method, re are at least the following two limitations. First, the Fitts' index of difficulty of traveling across the screen to quire the arrow widget at the end of a scroll bar can be up to 8 bits, which may take 2 seconds to complete. Second, to go to the scroll bar to move a document, even by just one line, takes the perceptual, cognitive and motor resources away from the main task, breaking the work flow.

With dual-stream input methods, one stream can be exclusively used for pointing and the other for scrolling. Two of the dual stream methods we studied were variations of a conventional mouse. One, called WheelMouse in this study, was a mouse augmented with a rolling wheel for scrolling (the Microsoft IntelliMouseTM, Fig. 1). The second device, labeled as JSMouse in this study, was a mouse augmented with an isometric joystick (an IBM TrackpointTM) for scrolling (Fig. 2). Both of these devices were manipulated with one hand. The third method, labeled as 2hand condition, used the same sensors as in the second condition, but with a different design: the isometric joystick was in the keyboard and was operated by the user's non-dominant hand (Fig. 3).

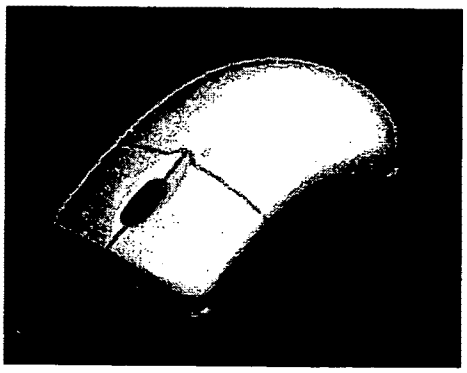


Fig. 1 Mouse with a tracking wheel, the IntelliMouseTM

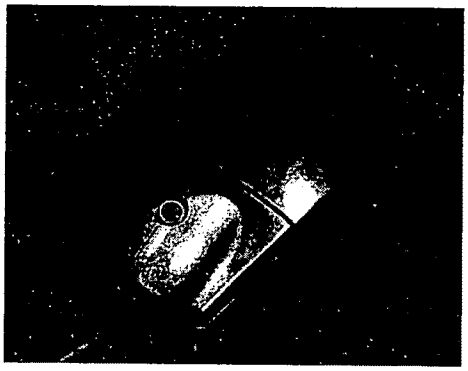


Fig. 2 Mouse with a Joystick (Trackpoint(TM))

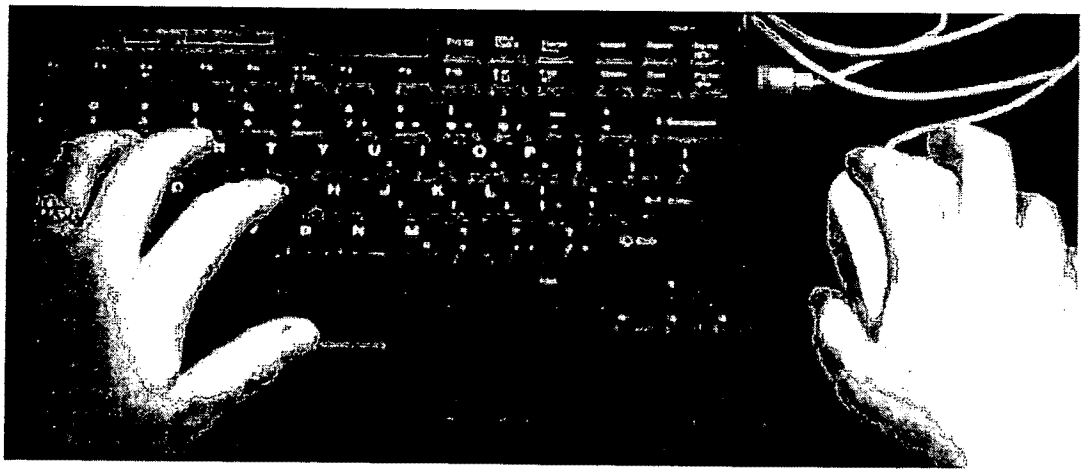


Fig. 3 Two handed system: an in-keyborad isometric joystick and a mouse

The Experiment

The goals of the experiment were: 1. Measuring performance and preference of the three dual stream input methods against conventional single stream input; 2. Comparing a rate control isometric joystick with a position control rolling wheel for scrolling tasks; 3. Contrasting the one handed and two handed methods.

he experimental task was to browse 10 web pages: scroll each page, and find and click on the target hyperlink in the page, which led to the next web page.

A total of 12 subjects participated in the experiment, with an order balanced within subject design. With each method, the subjects were first given one practice run which lasted as long as the subjects needed to explore all modes (in the cases of Mouse and WheelMouse). The subjects were then asked to performed two consecutive tests (10 pages each test) as quickly as possible.

Of the 12 subjects, all had extensive experience with using a mouse; five had experience with using the in-keyborad isometric joystick; all but one had no experience with the three dual stream methods. After completing the experiment, subjects were asked to rate each of the four methods.

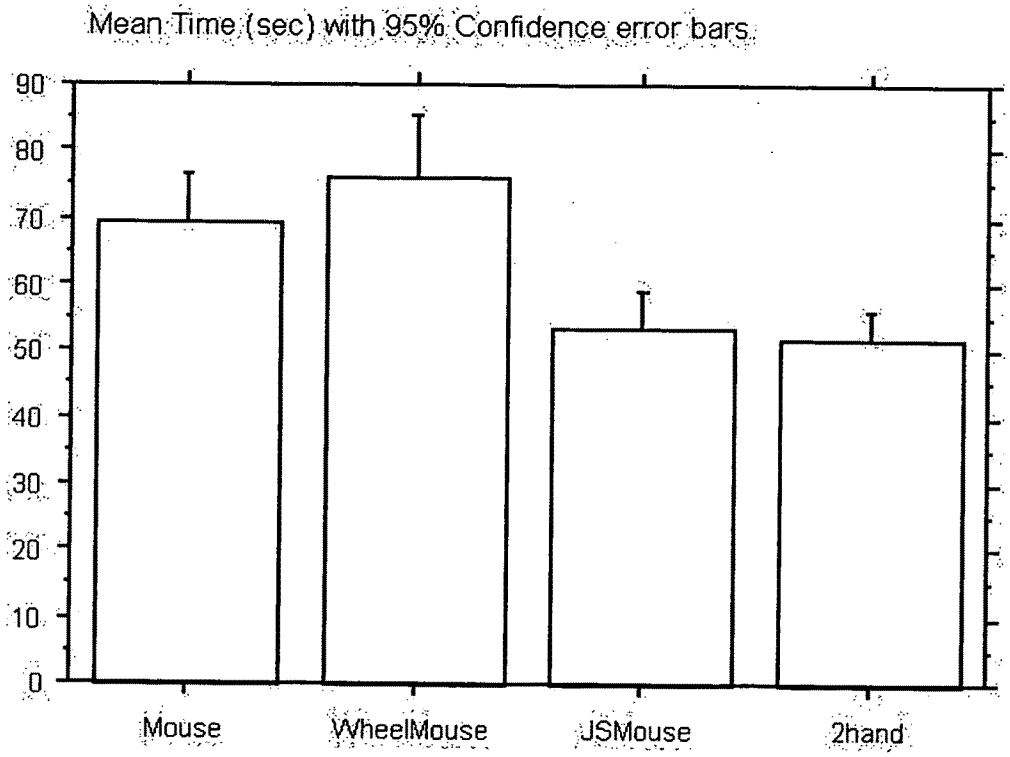


Fig. 4 Mean Completion time of web browsing task

Repeated measure ANOVA analysis on the results showed significantly different completion times among the four techniques ($F\ 3, 11 = 20.3$, $p < .0001$). As shown in Fig. 4, the JSMouse and 2Hand conditions were 22 and 25 percent faster, but the WheelMouse condition was 8.7 percent *slower* than the conventional single stream mouse. The difference between Mouse and WheelMouse conditions and the difference between JSMouse and 2Hand were not significant. All other pairwise comparisons were significant.