On Building a Light-Seeking Robot Mechanism

Stephen A Allen
POB 2281
Leucadia CA 92024

Tony Rossetti
1455 Arbor Av
Los Altos CA 94022

The idea of the Tee Toddler was born during the summer of 1976. We, as two undergraduate engineering students at Rice University in Houston, wanted to design a system using as much applicable electrical engineering as possible which could act on its own intelligence and which could also learn from its mistakes. We wanted to incorporate state of the art electronics and actually develop a piece of working hardware. As a three credit hour course for two semesters we designed and built a small "robot" car, the Tee Toddler.

The car is designed to track toward a shining light. It accomplishes this with the help of two processors: an on board Z-80 microprocessor which communicates with a PDP-11 minicomputer over a two way digital radio link. The source light the Tee Toddler searches for can be anywhere on the horizon. This light is detected by a rotating eye which scans a 360 degree view five times per second (see photo 1). There is also an ultrasonic sonar system capable of scanning simultaneously to the left and right of center to detect objects in the car's forward path (see photo 2). It can give ranges of up to five feet with 9 inch accuracy. The car has three forward and reverse speeds, five steering positions and a turning radius of five feet. Other standard equipment includes a front contact sensing bumper to detect objects which the sonar missed; a source light monitor to determine if the car is at its destination; a source light verifier to indicate whether the car has gone behind something which blocks the source light; whitewall tires and positive traction rear end. Most power requirements were met by regulating a 12 V rechargeable battery. The Tee Toddler is a closed loop system. It constantly updates its knowledge of where the light is and what obstacles are in the way; thus operating as a self-sufficient real time system. Great flexibility inherent in the two processor system allows for development of the car's intelligence. The programmer has lots of freedom in deciding how the car should deal with differing situations. This freedom in configuring the system between the computers and a moving object is the true beauty of the Tee Toddler. The duties of each processor are different. The on board Z-80 handles the car's reflex maneuvers; the PDP-11 makes both real time navigational decisions and can also generate a better path for the car to take on a second trip over the same obstacle course toward the light. The normal mode of operation is set up with the PDP-11 in control of the car via the radio. The Z-80 is operated in an

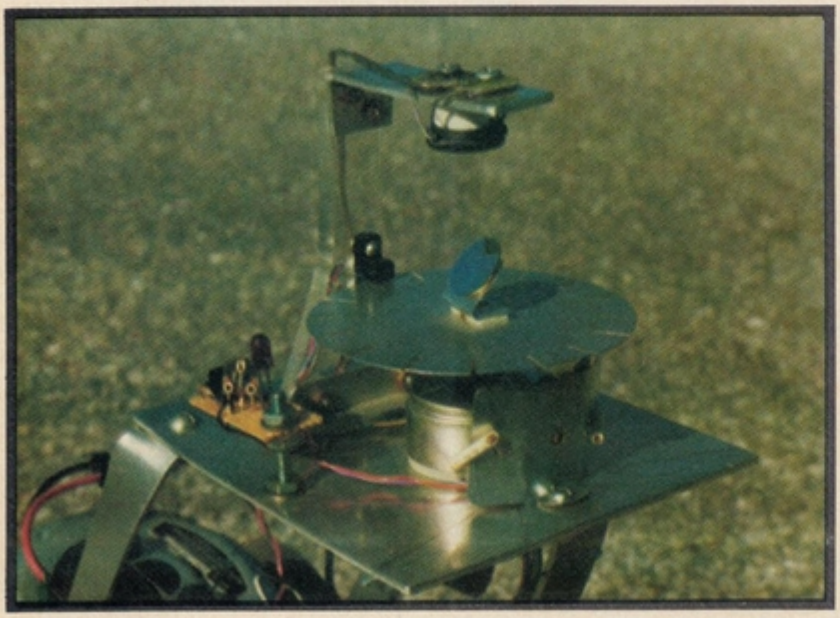


Photo 1: The primary sensor of Tee Toddler is this photoelectric horizon scanner. The "eye" mounted on a Plexiglas standard and metal bracket scans a 360° field. The flat mirror rotates at five revolutions per second deflecting light into the phototransistor eye through a 45° angle. The position of the mirror is resolved into one of 16 angular states by a slotted disk which passes through an optocoupler.

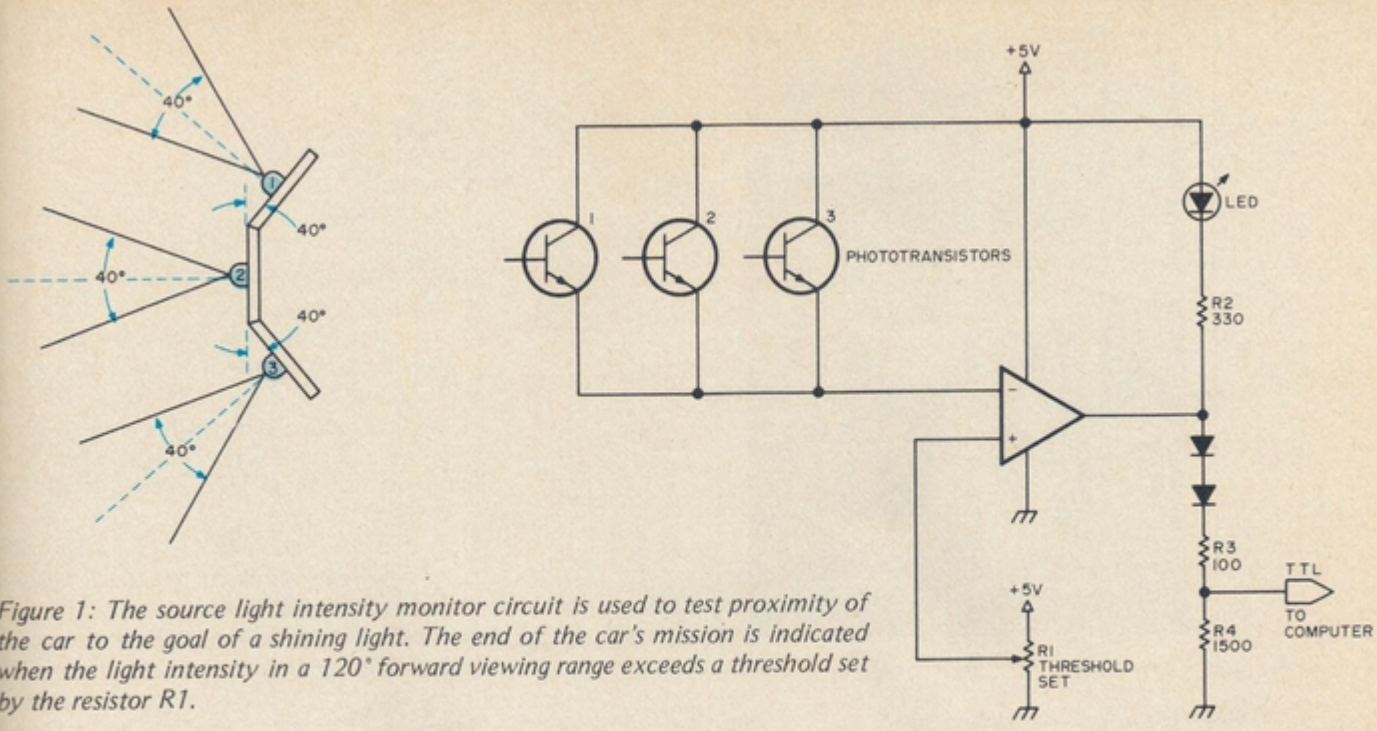


Figure 1: The source light intensity monitor circuit is used to test proximity of the car to the goal of a shining light. The end of the car's mission is indicated when the light intensity in a 120° forward viewing range exceeds a threshold set by the resistor R1.

interrupt mode. It stands by and records all course changes the car makes each time an obstacle is detected by sonar and also records each navigational correction made enroute to the light. In the case of collision or loss of the light, the Z-80 takes over control and remedies the situation before it returns control of the car to the PDP-11. At the end of the trip to the light the Z-80 dumps all the course change vectors it has recorded (steering setting and distance traveled) to the PDP-11 over the radio channel. The PDP-11 then determines a better path for the car to take over the same course on a second run.

Sensors: the Bumpers

The bumper is needed only in case an object is encountered in front of the car which is too narrow to be seen by the sonar. The bumper has two microswitches behind it, one on each side. It pivots in the center so that a contact on only one side will depress only one switch; however if the collision is head-on, both switches will be depressed. Two bits, one for each switch, are sent to the Z-80 computer.

Sensors: Source Light Intensity Monitor

The source light is the car's destination. The intensity monitor consists of three phototransistors which sense the intensity of light until the car is close enough to the source light (a foot) to stop; mission accomplished (see figure 1). An angle of 120 degrees is monitored, so the car must make

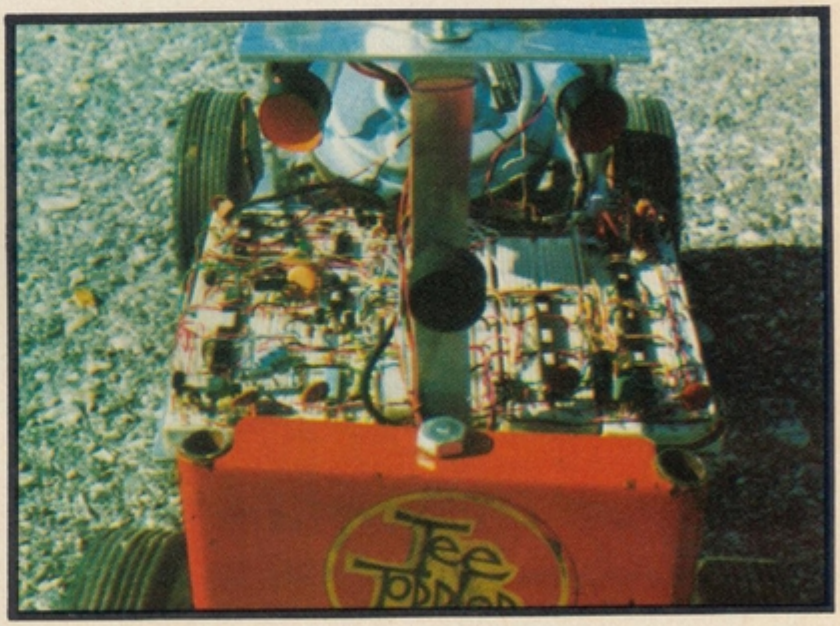


Photo 2: Tee Toddler's sonar system transducers are illustrated in this front view. The black object in the center of the picture is the sonar transmitter which emits periodic pulses of sound at 40 kHz. The two cup-like objects with red interiors (equally spaced to the left and right of the center of the picture toward the top) are the receiving microphones. The sonar drive electronics of this system resolves four distance states on each receiving microphone with a maximum range of about five feet and an accuracy of about nine inches.

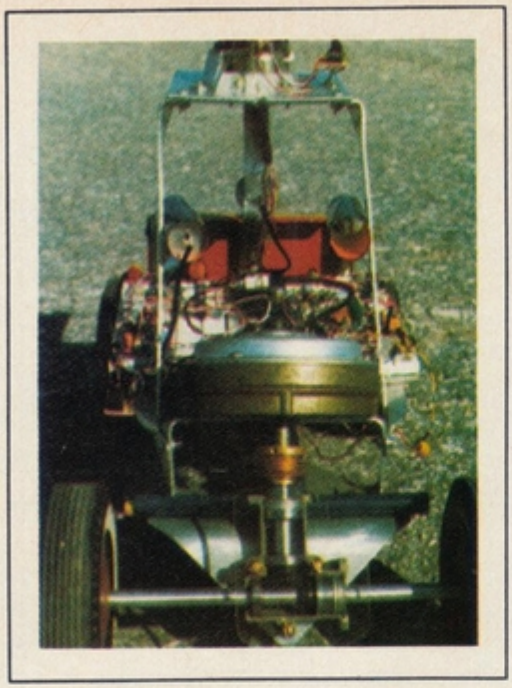


Photo 3: A view of the Tee Toddler car from the rear with the differential and drive motor visible. (The battery and rear deck have been removed for purposes of this photograph.)

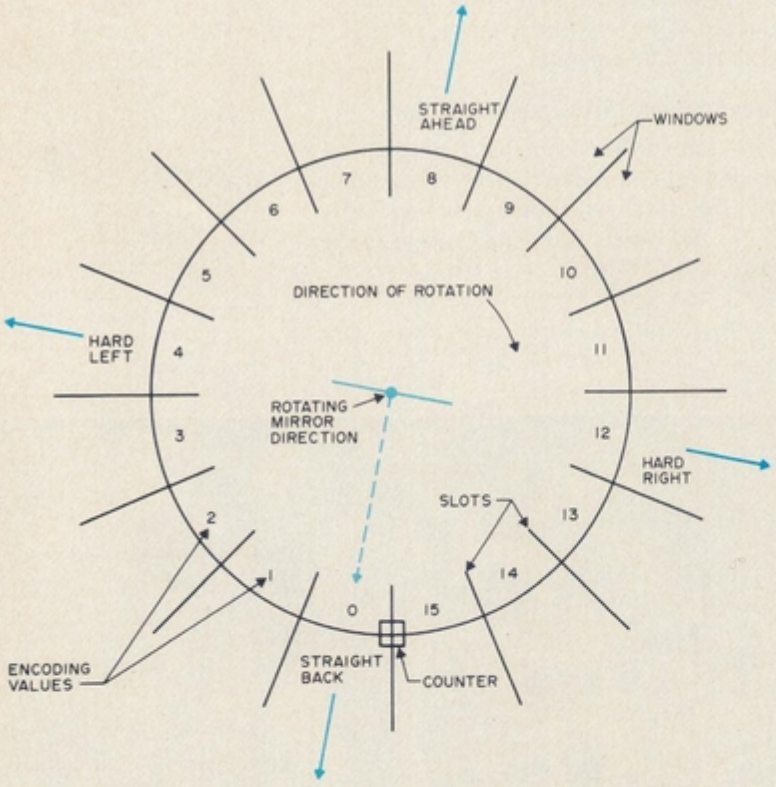


Figure 2: The direction of the light source relative to the forward direction of travel is measured by one of 16 angular states. The disk on which the main horizon scanning sensor's mirror is mounted has 16 slots which are sensed by an optocoupler which drives a counter. The counter is reset once per revolution of the disk by a separate sensor, so the angular states numbered 0 through 15 are sensed. When the photosensor detects the target light, the current state of the scanning angle is latched and can be read by the Z-80 mobile computer for transmission to the PDP-11 base computer.

its final approach moving in a forward direction.

Sensors: Rotating Eye and Source Light Verifier

The rotating eye scans a plane about 18 inches above the ground looking for a light source. Photo 1 shows the physical arrangement. It automatically adjusts its sensitivity for ambient light, much like the human eye. The response of its electronics is such that it can detect a penlight at 30 feet in a dark room. In a normally lighted room it is self-adjusting and can discriminate between two lights if one is about three times as bright as the other.

As the disk rotates clockwise, the 16 slots in its edge pass through an optical switch and are counted by a 4 bit counter on the main deck. At the instant the light is spotted during the disk's rotation, the count is loaded into a 4 bit latch to be read by the computer. For example, if the light is spotted straight ahead, the count is eight. Figure 2 shows the logical definitions of the 16 possible directions (a missing slot corresponds to the state when the mirror is aimed to the rear; the counter is reset to zero in this condition). Thus any erroneous counts caused by ambiguous light sources or reflections are wiped out each time the disk begins a 360 degree scan. Once a number is loaded into the latch it stays there until the light is spotted again and a new number is loaded. This reloading usually occurs once for each time the disk goes around. But if the light source suddenly becomes blocked by some object, the latch continues to hold the last number loaded even though there is no light being seen. To remedy this problem, a source light verification circuit is part of the electronics. This circuit sends a logical 0 to the Z-80 as long as the light is actually still being spotted and a logical 1 when it is not.

Steering Control

The steering system has five possible positions numbered arbitrarily 2, 3, 4, 5 and 6. 2 is far left and 6 is far right. The command from the computer (PDP-11 or Z-80) has 3 bits to specify these states. The number is converted to an analog voltage using a current sourced resistor ladder. The DC voltage enables a pulse width modulator which controls the two steering servos. The servos act in opposite directions on opposite ends of the front axle to turn the wheels. The pulse is sent at 67 Hz. See figure 3 for a block diagram of the steering control section. If the pulse is 1 ms long, the servos stay where

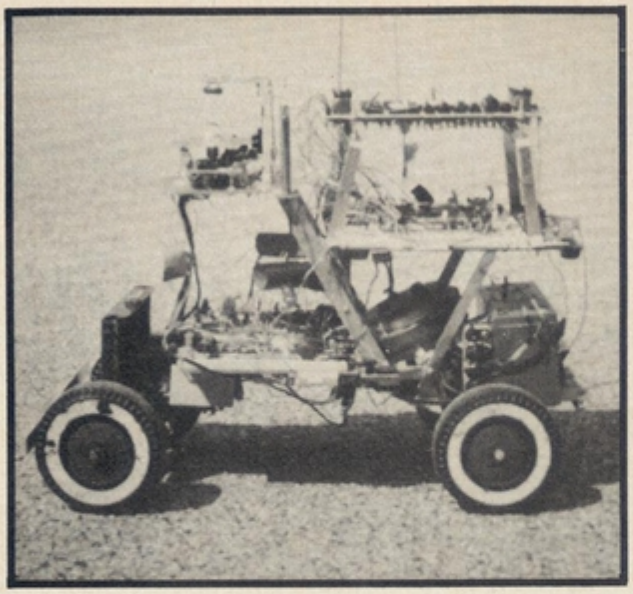


Photo 4: An overall view of the Tee Toddler taken from the side. Radio antennas and three levels of electronics on board are visible. The drive power source, a GeLi rechargeable battery, is at the right, with the front of the vehicle towards the left in this photograph.

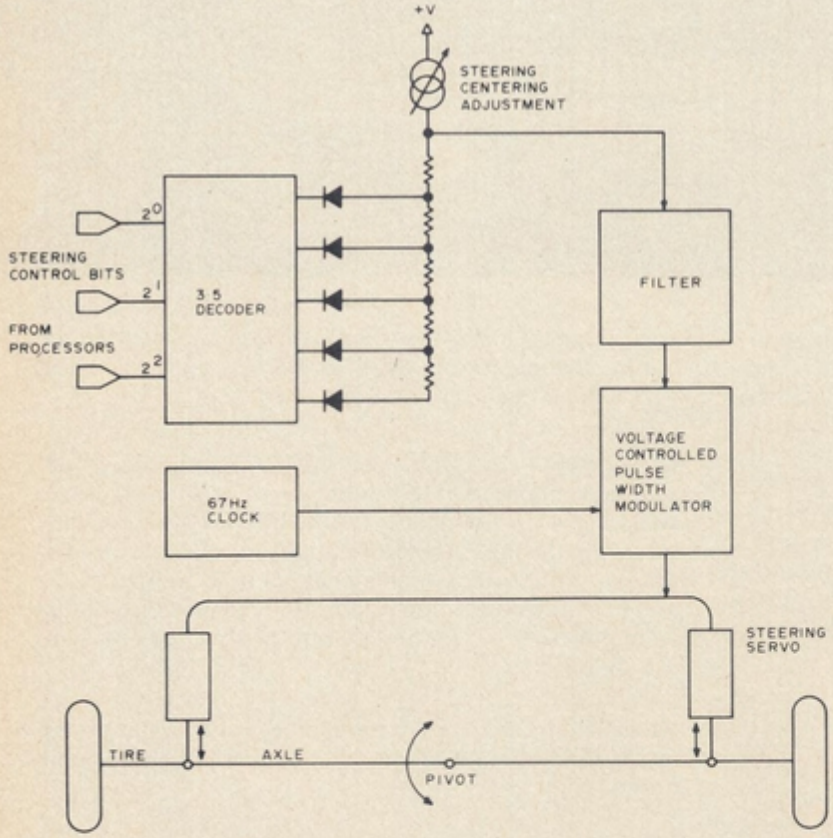


Figure 3: Steering system block diagram. Three bits from the Z-80 are decoded by a TTL decoder which implements a simple weighted resistor digital to analog converter. The output of the converter is filtered to prevent sudden changes and, in turn, sets the pulse width seen by the two servos. The servos are Heathkit radio control devices which have a 5 pound thrust and a 1.5 second full range response time.

they are; if it is longer or shorter, they move the wheels left or right. The servos are Heathkit radio control modules with five pounds of thrust. The change in pulse width seen by the servos is electronically filtered since the command from the computers can change from far left to far right instantaneously while the servos take about 1.5 seconds to pull the wheels from full left to full right.

Motor Control

The motor speeds are given by the numbers 0, 1, 2 and 3 which are decoded to stop, slow, medium and fast. There is also a forward and reverse bit, thus making a total of 7 motor states controlled by three binary digits. The motor control pulse width circuit works the same as the steering control circuit, except that the pulse change is not filtered. The motor is a 0 to 13 VDC "pancake" motor with a built-in 25:1 gear reduction. The rear axle differential gear ratio is 1:1. The motor's speed is controlled by the pulse width of the 12 V 700 Hz pulses being sent to it by the control circuit. Forward and reverse directions are controlled by a relay. The motor draws a current of about 1/2 A when the car is cruising at 1 mph (1.6 kmph). Since the motor is a highly reactive load to the sharp edges of the control pulses, the motor is optically isolated to eliminate interference with the logic circuits of the on board Z-80 system. Power amplification to drive the motor is accomplished after the isolator.

Sonar System

The sensing of objects in the car's path was originally intended to be done with light. This is difficult since objects with different textures at the same distance from the car would reflect different amounts of light. Pulsed infrared did not have the necessary intensity, and radar was ruled out because it would detect only metal objects. The existence of the National LM1812 sonar integrated circuit was probably the major factor enabling use of this sensor system. With this system the car is able to distinguish between obstacles to its left or right and can navigate between obstacles spaced only slightly farther apart than the car's width.

The sonar unit on the car transmits a 1 ms pulse at 40 kHz every 10 ms from a transducer mounted in the center of the car's front end (see photo 2). The echo is received separately on the right and left by two receiving transducers. Since there is

really only one sonar transceiver, the receiving transducers are multiplexed with field effect transistors to the receiver for three cycles of transmit and receive each. The count of the elapsed time between transmit and receive for each cycle is also multiplexed into the left or right output latch to be ready by one of the computers. A block diagram of the sonar system is shown in figure 4.

Since sound in air travels at about one foot per ms and a pulse is transmitted each ten ms, the maximum range is about five feet, which proves quite adequate. If no sonar echo is received, then the output latches are automatically loaded with the count 7, the maximum range possible. This is decoded by the computers as no object and thus no path correction is made. The sensitivity of the receiver is adjustable with a trimpot on the main deck, thus allowing different distance resolutions.

Sonar Ranging Terms: Derived Quantities to Be Found

θ = Angle of object relative to forward direction.

T_O = Time of transit out to object (equivalent to a distance). T_O is one side of both left and right path triangles.

Note: Using the law of cosines, and the two measurements, the algebra gives two equations (left and right signal path triangles) which can be solved exactly for two unknowns (T_O and angle θ).

Sonar Ranging Terms: Measured Quantities

B = Half the total distance between the two receivers. This forms one side of the triangles used with the law of cosines.

R = Measured transit time, transmitter to object to right receiver. $R = T_O + T_R$

L = Measured transit time, transmitter to object to left receiver. $L = T_O + T_L$

Sonar Ranging Terms: Miscellaneous

$90-\theta$ = Included angle used in law of cosines applied to left triangle.

$90+\theta$ = Included angle used in law of cosines applied to right triangle.

T_L = Time of transit back from object to left receiver (equivalent to a distance). T_L is one side of left signal path triangle.

T_R = Time of transit back from object to right receiver (equivalent to a distance). T_R is one side of right signal path triangle.

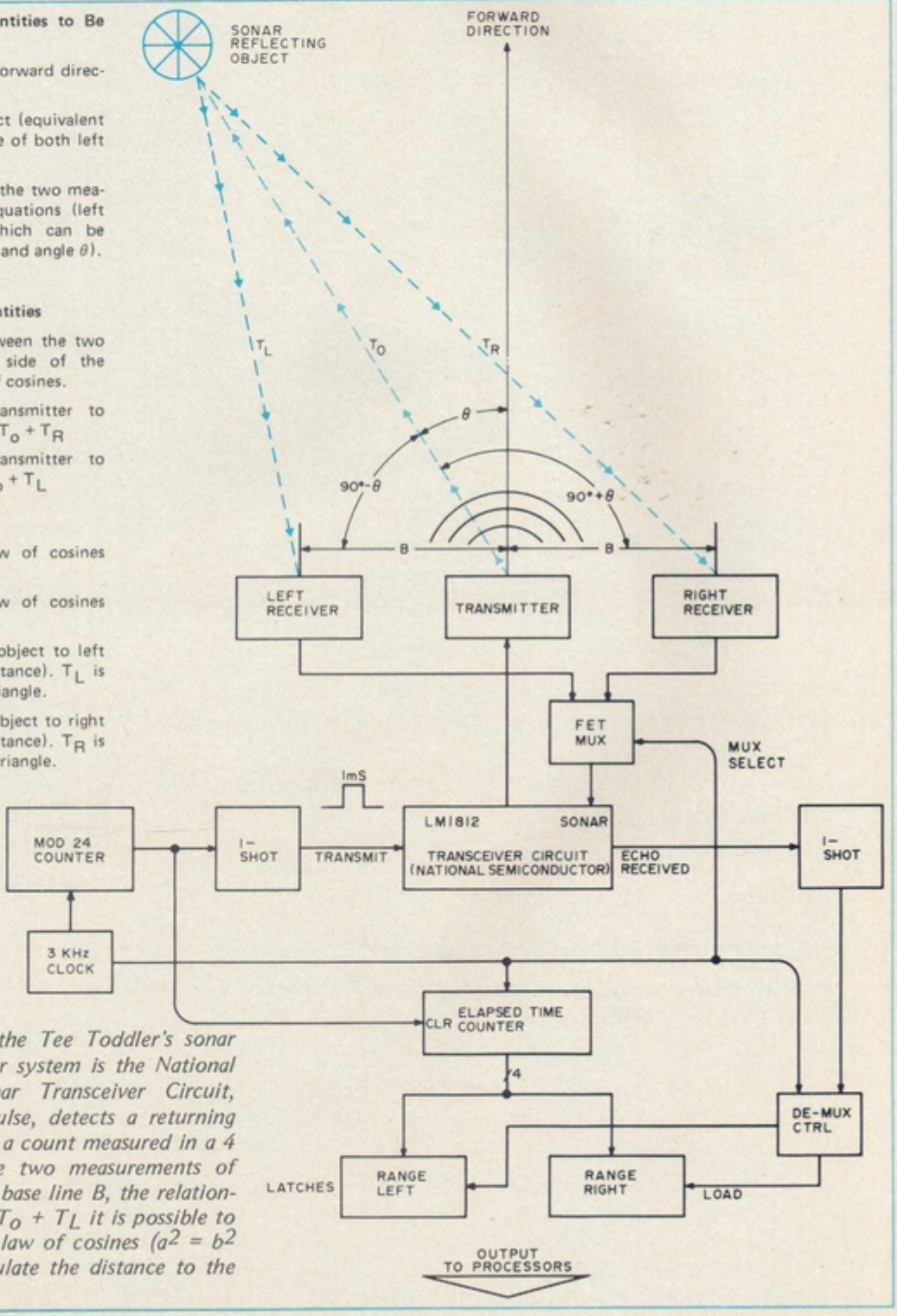


Figure 4: Block diagram of the Tee Toddler's sonar system. The heart of the sonar system is the National Semiconductor LM1812 Sonar Transceiver Circuit, which generates the sound pulse, detects a returning sound pulse and thus controls a count measured in a 4 bit range counter. Given the two measurements of distance R and L, the known base line B, the relationships $R = T_O + T_R$ and $L = T_O + T_L$ it is possible to apply the plane trigonometry law of cosines ($a^2 = b^2 + c^2 - 2bc\ COS\ (A)$) to calculate the distance to the object T_O and the angle θ.

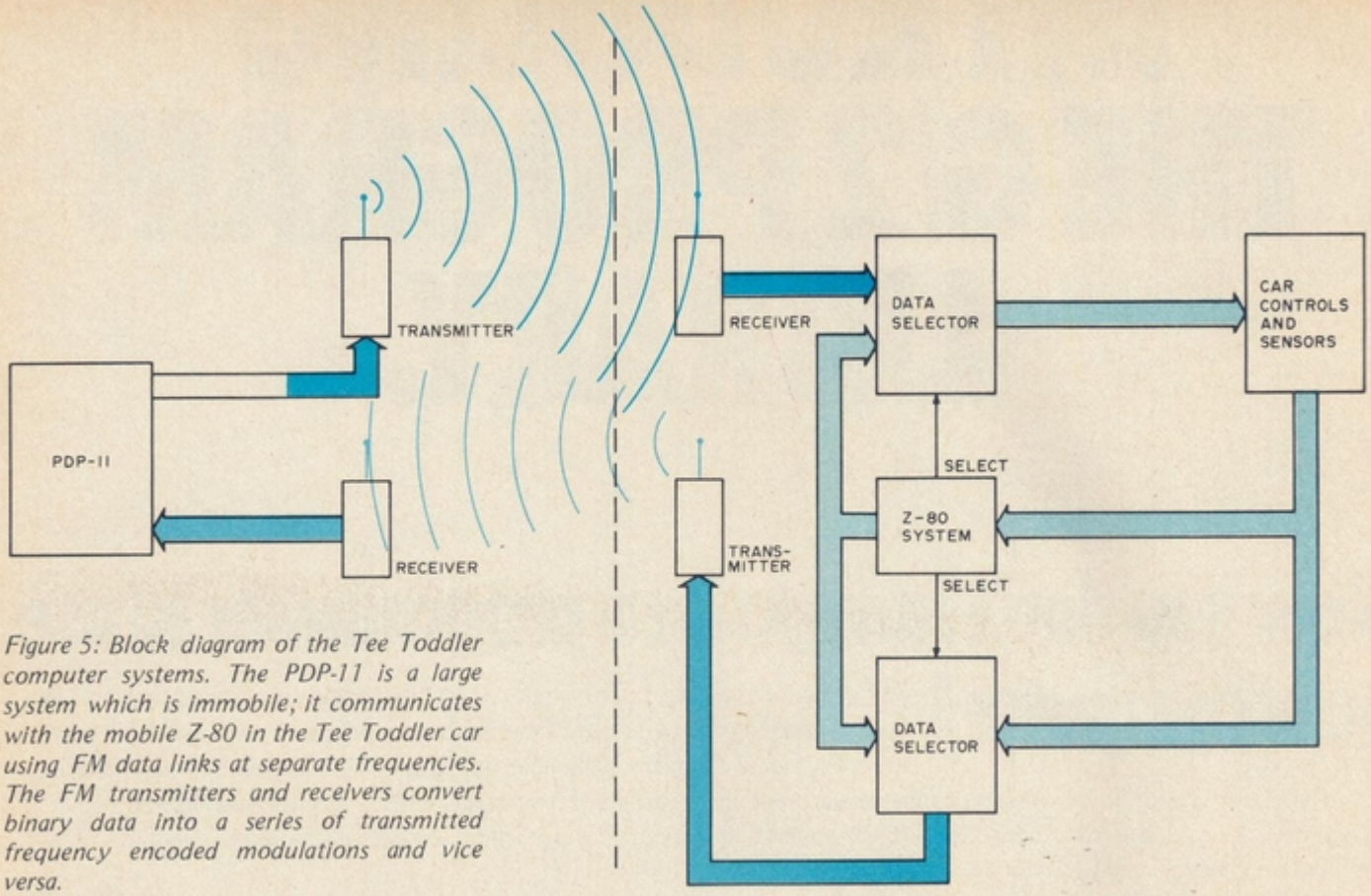


Figure 5: Block diagram of the Tee Toddler computer systems. The PDP-11 is a large system which is immobile; it communicates with the mobile Z-80 in the Tee Toddler car using FM data links at separate frequencies. The FM transmitters and receivers convert binary data into a series of transmitted frequency encoded modulations and vice versa.

This type of sonar system is really a minimal one. Doppler shift detection could also be accomplished fairly easily to allow determination of speed of a moving obstacle. Echo amplitude analysis would also be worth investigating since it would help solve the problem of echo frame overlap; such overlap exists when an echo from the previous sounding returns late, after bouncing off a far away object, resulting in two echos for the current frame. The strongest of the two (or more) echos should be taken as the true one. The Tee Toddler's system triggers a 9 ms oneshot on the first echo thereby ignoring all secondary echos.

Radio Data Links

An encoder transforms parallel bits into serial tones to be transmitted over a frequency modulated channel. One channel is from the Tee Toddler car to the PDP-11 computer at 96 MHz. The other channel is from the PDP-11 computer to the car at 450 MHz. The serial data encoder and transmitter at the PDP-11 base station are essentially identical to the car's versions except for the number of bits per word of data. A two-tone modulation system is used. This means that each binary state is encoded into one of two different frequencies for transmission. At the modulator a logical 1 is represented by a 2500 Hz signal and a logical 0 is represented by a 1900 Hz signal.

The receiver and data decoder accepts the string of audio tones from the FM receiver, decodes them into 1s and 0s using phase locked loops and converts back to a parallel data format. While our prototype did not use standard circuitry, a standard asynchronous serial communications discipline such as that provided by a UART or ACIA would work well in this application.

Power Sources

The power for most circuits is derived from a 12 V 4.5 Amp-Hour GeLi cell rechargeable battery. The battery was drilled and tapped at 8 V to power a 5 V regulator for the TTL circuits and for the Z-80 microcomputer. The steering servos required their own set of four penlight batteries (also rechargeable), and the 1702 read only memory holding the Z-80 program required a separated −9 V supply, derived from three parallel transistor radio batteries. This power supply system is capable of running the car for several hours before any recharging is necessary.

Computer Control

As has been mentioned, the car is con-

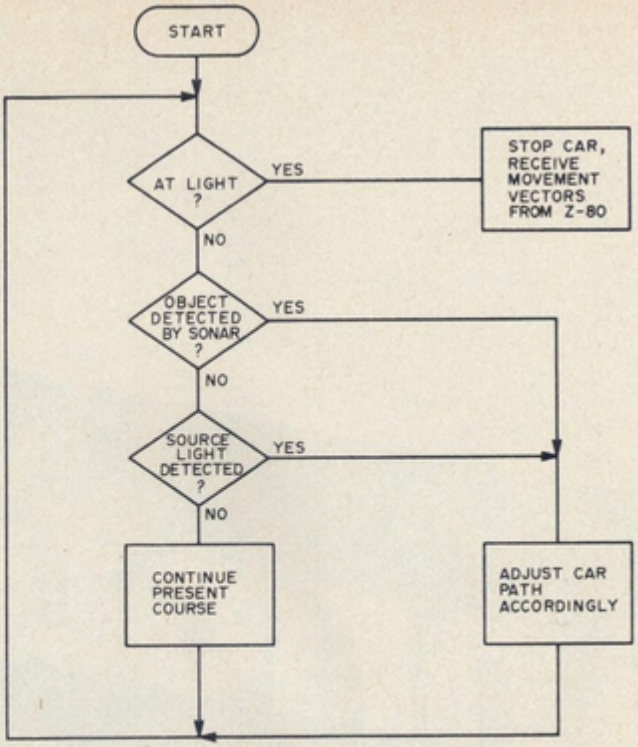
trolled by two separate computers. The communication paths between the computers and the car are shown in the system diagram of figure 5. In actuality, only one of the computers can communicate with the car at a time. This is the case for several reasons. First, there are only three control inputs to the car to make it operate. These are the speed, direction and steering controls. Since these inputs can originate at either computer, a multiplexing scheme had to be used. Second, only the PDP-11 actually makes decisions based on sensor information from the car. The Z-80's control of the car's movements is more like a reflex action, in that it performs a canned routine when invoked by the car's sensors. Last, the functions are separated to facilitate the transition to a total on board control system, since the PDP-11 can be replaced easily by another on board microcomputer.

The motivation behind this configuration is based on several criteria. Since part of the system was going to be standing alone, some of the major considerations were power consumption, various power supply requirements and ease of operation. With all these considered, it was decided that a Z-80 with its single 5 V power supply requirement and single phase clock was a logical candidate. The 16 bit PDP-11 was used because it could do computations at a greater speed than the 8 bit Z-80.

The on board microprocessor has several functions associated with the control of the car. One function is to supervise all data and control channels to and from the car. In other words, it has the responsibility of deciding whether the PDP-11 or the Z-80 is going to control the movements of the car and which of the two computers is going to receive the information from the car's sensors. The routing of these different channels of information is accomplished by the use of data selectors. The Z-80 controls the data selectors such that the information is routed to the correct device at the correct time. Information coming in to the car to control its movements comes from either the PDP-11 or the microprocessor. It comes from the PDP-11, over the radio link, if the car's sensors indicate one of the following conditions:

- The car has reached the source light.
- An object has been detected by the sonar system on either the left or the right.
- The car has spotted the source light.

The PDP-11 then analyzes these conditions according to the hierarchy of importance, as is shown in the decision tree of algorithm 1, and then communicates to the car the appropriate movement corrections to make. Control information to the car originates from the Z-80 when one of the following car sensor conditions arises:

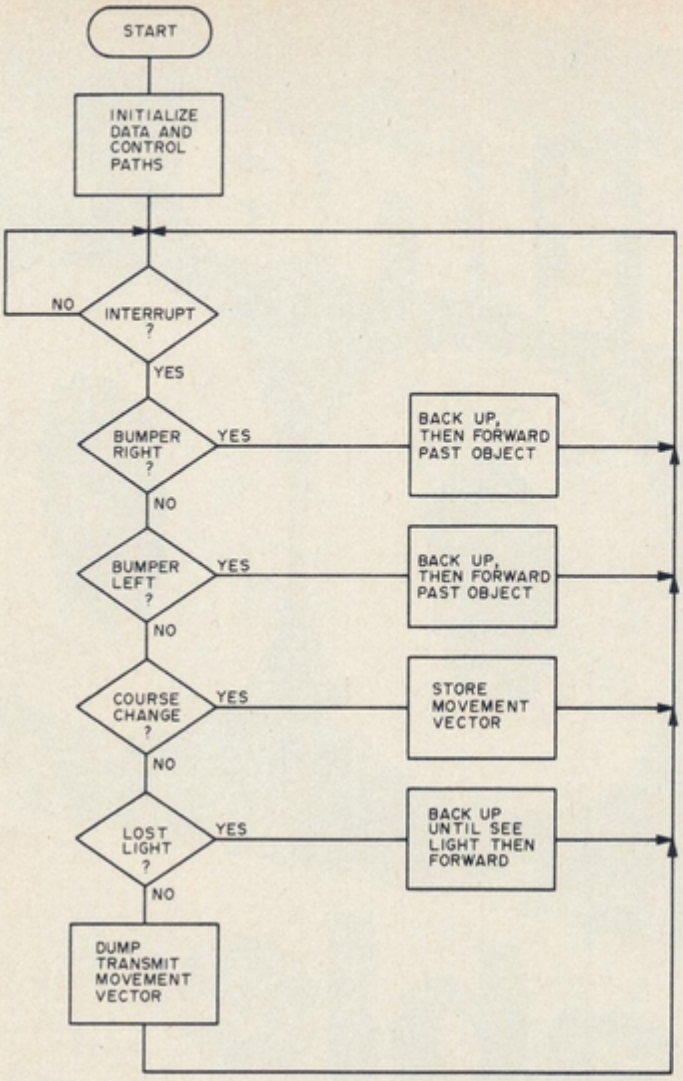


Algorithm 1: The base computer's executive program in outline form. This decision tree is executed in the PDP-11 each time a car sensor word is received. If any of the tests results in an affirmative answer, the program executes a routine designed for that specific state. Each routine takes into account past information of where the light was spotted. The sonar detection routines also take into account any objects which have recently been passed. These things are considered so that the car proceeds in the direction of the light and does not collide with any objects while moving in reverse. There is no specific way to stop the system except by interrupting the PDP-11 and issuing a control word to the car to stop it.

- Contact with an object has been indicated on either the left or right side by the front bumper.
- The car has lost sight of the source light.

The microcomputer controls the movements of the car if either condition is met and then gives control back to the PDP-11 when it has finished its corresponding task.

Another function of the on board microcomputer is to store all movement vectors associated with the car's path. These vectors indicate the steering angle, the direction of travel and the length of travel of the car. Therefore, when the car changes direction or steering angle, a vector is stored in memory which correlates to how far the car traveled at the previous setting. Thus, when the task of finding the light is accomplished, the on board memory contains all the different moves the car made to reach



Algorithm 2: The mobile computer's executive program, in outline form. In the initialization procedure the Z-80 sets up the memory areas, resets the wheel rotation counter and sets the data selectors for the PDP-11 to control the car and the car sensor status word to be transmitted to the PDP-11. The processor then awaits an interrupt. When the Z-80 is interrupted, the tests are executed in this order. After each routine is completed, control is returned to the PDP-11.

the light. With this information the car has a new path calculated for it by the minicomputer. This path allows the car to drive to the light from the same starting position without the use of any of its sensors and without having to maneuver around a single object, since it already knows where they are located. In a very loose sense then, the system has learned about its environment and used this knowledge to improve its proficiency at the task of finding the source light, much like a mouse in a maze.

The last major function of the microcomputer is to pass the movement vectors to the PDP-11 once the car has reached its destination. This is accomplished by having the minicomputer issue a "dump" command signal over the radio to the Z-80. Using a handshake system, the Z-80 sends the movement vectors to the PDP-11 one at a time. The decision hierarchy of the Z-80 program is shown in algorithm 2.

PDP-11 Base

The PDP-11 minicomputer is the actual brain of the system. It has the ability to decide where to move the car in order to approach the light and yet avoid objects on the way. Inputs to the PDP-11 come from either the car's sensors or the microcomputer memory. If the inputs are from the car they indicate the current status of the sonar left and sonar right sensors, the 360 degree rotating eye and the source light intensity monitor. These are processed according to the following hierarchy. First the source light intensity monitor signal is checked to determine if the car has reached its destination. This indicator is checked first because it will indicate if the total task has been accomplished. If this condition is true, the PDP-11 computer sends a message to the car telling it to stop and telling the Z-80 to start unloading its memory of movement vectors. The handshake system used is initiated by the car informing the PDP-11 base computer that the car has reached the light. The minicomputer then informs the microcomputer to start unloading the memory, at which time the minicomputer checks each incoming vector to determine if it is the last. If not, the PDP-11 asks for another vector to be transferred. This continues until all vectors are transferred.

Second, the sonar sensor inputs are examined to see if any objects are being detected. If an object is detected, then a special routine analyzes the situation according to which side the object is detected and how far away it is. If the object is far enough away for the car to maneuver around it without having to back up, then the PDP-11 commands the car to steer to the left or right, whichever is appropriate, to avoid the object which is in the way. If the object is detected by both sensors, then the side which detects it as being closer overrides the other. In the event that the distance measurements are equal, the computer arbitrarily chooses the right side as having a higher priority. Obstacles detected at a range too close for the car to maneuver around while proceeding forward cause the car to back up. An obstacle detected at a very close range on the right causes the car to back up. However, the steering position for this movement depends on whether the car was steering to the left, right or center. If the car was proceeding to the right, then it must know, from a previous sensor reading, that the source light is to the right. If this is the case, the car backs up

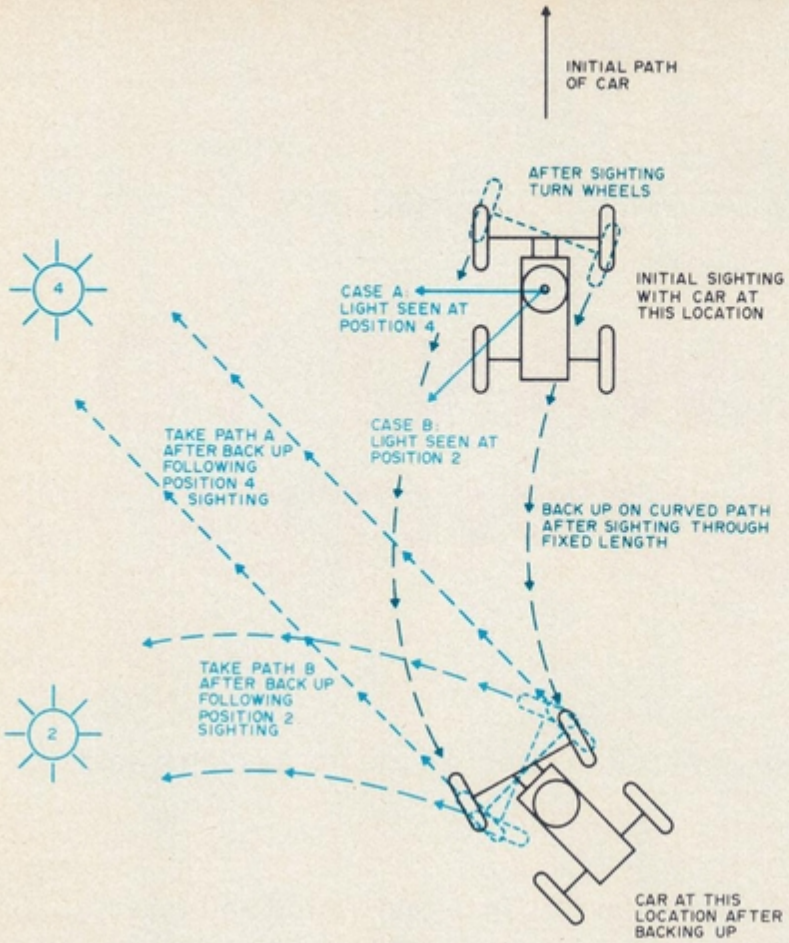


Figure 6: This diagram illustrates paths the car can make when confronted with typical situations. Assume the car is moving straight and forward and detects the light in either the number 2 or 4 position. The PDP-11 has control and moves the steering to the right and backs up for a certain distance. If the light was spotted in position 4 the car proceeds along path A with the steering set back to the center. But if the light was spotted at position 2 the car moves along path B with the steering set to the left. If the light had been spotted at position 1, almost directly behind the car, the maneuver would also have been along path B. However, since it would not be able to approach the light at a very straight angle, it would have to back up again and set the steering for a more direct path.

with the steering set to the left. After backing up for a certain amount of time, the car changes the steering to the right and proceeds forward. This causes the car to maneuver around the obstacle and also sets it on a better path to the source light. In effect, the car has used past knowledge to evaluate the present situation. If the steering had been to the left originally, then the computer would set the steering to the right. The car then proceeds in reverse for a given amount of time and then changes its steering position to the left and proceeds forward for an additional period of time. This action makes the car maneuver around the object and along a better path. An original steering position in the center again causes the right side to override the left. Although all these controls and decisions are handled by the minicomputer, the results do incorporate the use of the on board microcomputer. The microcomputer is used to store all movement vectors pertaining to all direction or steering changes. This is accomplished by having the PDP-11 computer issue a course change signal to the Z-80 at the same time it issues the new control word to the car. The Z-80 on board computer then stores the previous movement vector in memory, then returns control to the minicomputer.

The third sensor readings used by the PDP-11 are the values from the rotating eye. These sensors indicate where the light is located with respect to the current position of the car as shown in figure 2. Basically, the world, as the car sees it, has been divided into 16 windows, each allowing a different view of the horizon. A number has been assigned to each separate slice, thus giving an easy identification and recognition scheme for evaluating the position of the source light. For example, the semicircle in front of the car has nine windows associated with it, four to the left, four to the right, and one for the center. Thus, since straight ahead has been declared as having a value of 8, then the far left becomes 4 and the far right becomes 12. Therefore, by examining the value, the computer can tell where the source light is and then adjust the path to proceed in that direction. If the light is spotted in the forward semicircle between values 5 and 11, the course adjustments are quite straightforward. The steering is merely positioned so as to point the car in the direction of the light.

However, if the light is spotted in the aft semicircle or to the extreme sides, a different approach must be taken. Instead of having the car do a complete 180 degree turn, we decided to have the car perform several backward and forward movements. By doing this we reduced the possibility of contacting objects by reducing the space needed to perform the maneuvers. For example, if the light is detected in the number 1 through 4 windows, the computer backs the car up with the steering set to the right. The distance it backs up depends on which window the light was spotted in. For example, if it was spotted at position 4, the extreme left, then the car would back up far enough so that when it stopped it would be facing directly toward the light and could then proceed in a straightforward direction (path A in figure 6). If, however, the light had been spotted more to the rear, the car would back up a bit further and then have the steering set to the left position and proceed forward. The exact opposite action would have taken place had the light been

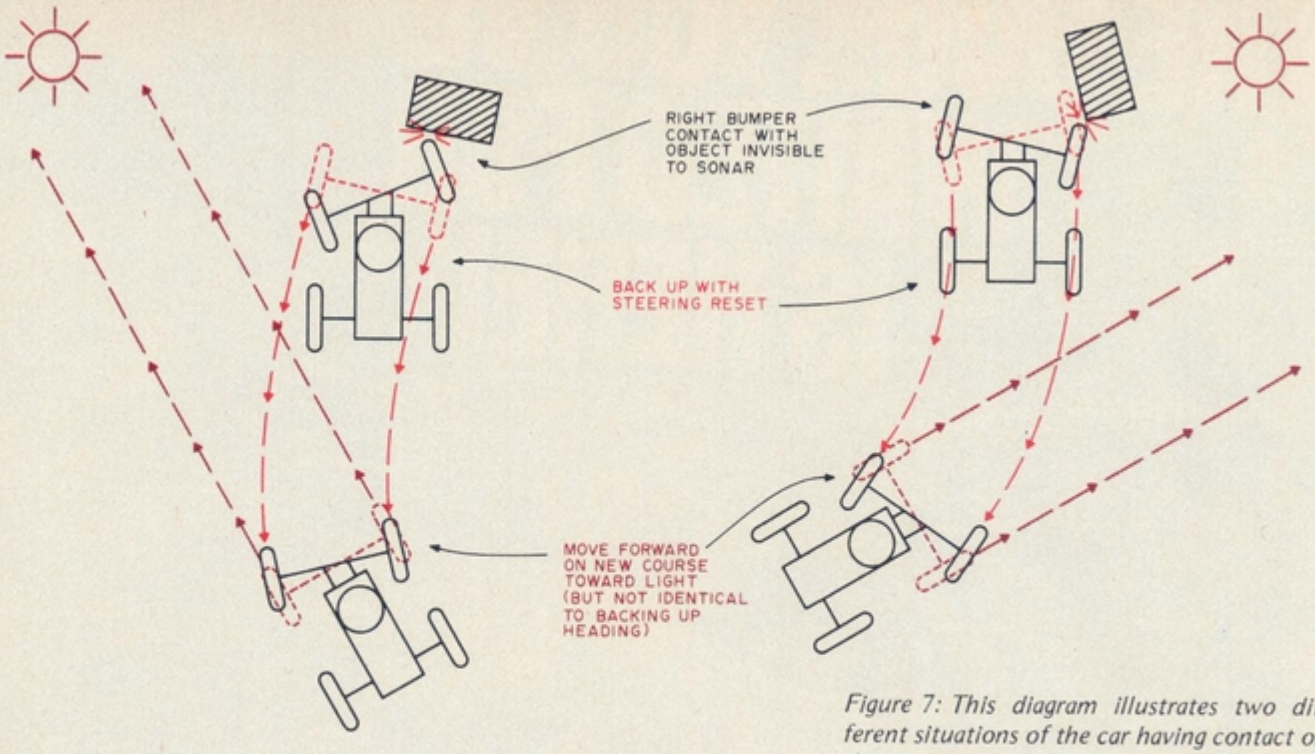


Figure 7: This diagram illustrates two different situations of the car having contact on the right with an obstacle. The obstacle size is exaggerated for the sake of illustration. The diagram on the left depicts the car steering to the left toward the light when it strikes the object. The Z-80 takes control and adjusts the steering to the right and backs the car up a certain distance. The steering is then set to the center position and the car proceeds forward. On the right, the car is proceeding to the right toward the light. Here the Z-80 sets the steering to the left, backs the car up then adjusts the steering to the center and proceeds forward. In each instance the car maneuvers around the obstacle and on a path toward the light.

spotted on the opposite side. Once again the on board computer would have been interrupted to store all course changes associated with the maneuvers. The problem of running into objects while proceeding in reverse is actually of minimal concern due to this method of reaching the light when it is spotted in the rear. If there had been an object there it presumably would have been detected and the path adjusted accordingly. However, this adjustment would not have been made without taking into account where the light was being spotted. Since this is the case, it is not possible for the car to be turning left or right in a forward direction if the light is actually behind the car. If, however, the light has not been spotted yet, then it is feasible that the car can run into something in its reverse move, since no attempt is being made to look back into the movement vector memory to determine if an object has just been maneuvered around.

The final function of the minicomputer is to stop the car once it has reached the source light. This is accomplished by detecting the source light intensity monitor bit as it changes to the active state. Once this occurs the minicomputer stops the car and at the same time initiates the handshake operation with the on board computer to start the transfer of the movement vectors. The minicomputer stores these vectors as they come over the radio until all have been passed. The last vector is actually a null vector, or all zero, which indicates all vectors are transferred. The minicomputer now does one of two things. It can automatically plot out a new course for the car to take, or it can display the vectors graphically on a screen. With the latter method the user is able to see all the moves and recalculate a new path himself based on his visual perception of the path taken. The automatic method is simply a sequential analysis of the vectors by the computer. If the car makes a move in reverse the computer assumes that either an object was detected or the light was spotted behind the car. In either case the computer adjusts a move previous to this occurrence, thus allowing the car to anticipate the upcoming situation and act in accordance with the situation. By adjusting these movements prior to detecting the need to reverse direction the computer has eliminated this need altogether and has thus "curved out" the path, making broad sweeping turns as opposed to jerky forward and backward movements.

Microcomputer Functions

The on board Z-80 computer provides the reflexes and signal control for the whole system. In the event that the car hits a thin object which is not detected by the sonar, a reflex action is invoked, much like a human response to a given stimulus. The various reflex actions this computer controls are: loss of sight of the light and touch stimulus from either sides of the bumper. To initiate a microcomputer routine for either the reflex actions or the control functions, one of the following interrupt inputs must become active:

- A signal from the front bumper.
- A signal indicating loss of the source light.
- A course change.
- A request to dump the movement vectors.

All these signals are ORd together, thus enabling any one of them to initiate an interrupt. When the Z-80 is interrupted it interrogates an external buffer to determine which condition caused the interrupt. The program (see algorithm 2) then checks each bit, one at a time, to determine which one is active. If more than one is active, it only processes the first one checked which is active. If none of the lines is active, then the program defaults to the dump line being active. Upon determining which stimulus is active the program executes a specific routine for that particular interrupt.

The bumper right and bumper left routines are essentially the same except for the steering positions being reversed. If contact is detected with the right side of the bumper, the Z-80 receives an interrupt and the car automatically backs up. Figure 7 illustrates the bumper reflex. The direction in which it backs up depends on the direction it was travelling when it collided with the object. If the steering was set to the right, then it must have previously detected the source light to the right (see right illustration in figure 7). Then, in order to maintain this general direction, the car backs up with the steering set to the left. After backing up for a certain time the car sets the steering to the right and proceeds forward past the object and toward the light. Although this setting is not a direct heading toward the light it is in the general direction and it has avoided the object.

If the direction of travel was to the left, (left illustration, figure 7) then the car backs up with the steering to the right and then proceeds forward with the steering set to the center. All steering actions would simply be reversed for a contact on the left. Therefore it is easy to see that all the reactions to the stimuli are preprogrammed and always net the same result, thus they exhibit a reflex action.

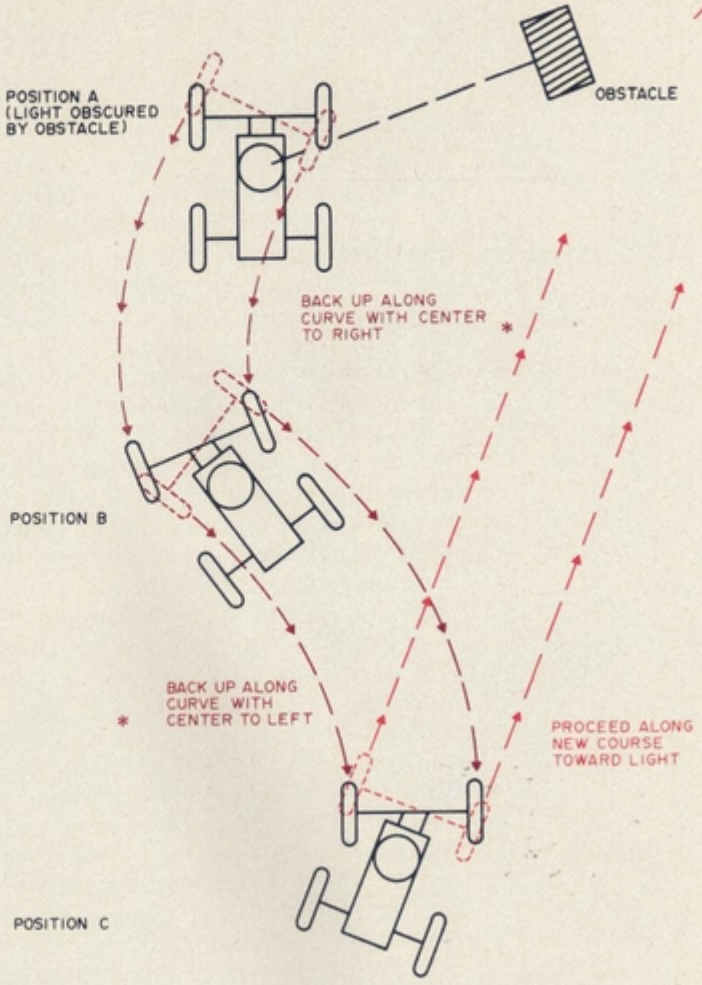


Figure 8: This diagram illustrates the Z-80 reflex for the case in which the car has lost sight of the light at position A. The Z-80 instructs the car to do a reverse S turn. First it adjusts the steering to begin the turn and travel to position B. Then it adjusts the steering to have a center of curvature to the left and continues reverse travel to position C. At the end of the path, the steering is again set to the appropriate position for a course towards the light and the car resumes forward travel without its goal being obscured by the obstacle.

The other reflex action is quite similar (see figure 8). If the car is travelling in any direction and loses sight of the source light, then apparently what has happened is that an object has come between the car and the light, thus obscuring the car's "vision" as at position A in figure 8. Although the car knows the object is there, it cannot detect

its exact location. What the car does then is back up with the steering in the same position, then after a certain time it changes the steering to the opposite side and continues to back up. It continues backing for another period of time and then sets the steering to the center and proceeds forward. This procedure causes the car to move to a new position where it can again see the source light. This again exhibits a certain reflex action controlled by the Z-80.

Since these reflex routines involve many adjustments to the car's path, it is necessary to record all of these separate movements. Therefore, at the end of each segmented move the routines call on the course change routine to record the current wheel rotation count, direction of travel and the steering position. This course change routine can be called on from either the Z-80 through another routine or directly from the PDP-11. The routine reads a buffer register which contains the current number of wheel rotations at this particular steering and direction position. Once this vector is stored in memory, the routine resets the wheel counter to zero in order for it to count the correct number of revolutions at the next steering and direction setting. The last function of the Z-80 is to transfer all the movement vectors from the on board memory to the PDP-11. As has been discussed, the Z-80 responds to a request from the PDP-11 by sending the vectors on a last in first out basis, one at a time in correspondence with the handshake system. Once all vectors are passed the Z-80 reinitializes the car and passes control to the minicomputer.

After any of these routines has been processed the Z-80 returns control of the car to the PDP-11.

Computer Design Specifics

The microcomputer designed for this robotic application is equipped with only the bare essentials. The total system consists of 256 bytes of programmable read only memory, 1 K bytes of volatile programmable memory, three bidirectional IO ports, one Z-80 microprocessor and one 8 bit line driver. The read only memory is a 1702 UV eraseable part which contains the program. The programmable memory is made up of eight 2102 parts. The IO chip is an Intel 8255 and was chosen because of the number of ports available.

When designing a dedicated system like this, one must keep in mind that the probability of it working the first time is very nearly zero. Therefore, care must be taken to make the system as easy to debug as possible. This system was designed with this in mind and thus several additional functions were included in the design. A reset switch is installed on the computer board to aid in checking different functions of the system under the same circumstances. A single step switch is also located on the board. By using this, one can step through the program and examine different signals to determine their validity. The line driver was installed specifically to allow the examination of the data lines. Included in the design are provisions for the addition of 1702 memory chips up to a total of 1 K bytes of program memory. Since the system was of a prototype nature it was built on a perforated board and was wire wrapped. Care was also taken in the use of the PDP-11. Before the final application program was written several simple test programs were written which checked out the two way radio links and the responses of the car to commands. With the test programs it was possible to enter commands at a terminal and control the actions of the car in a remote control fashion as well as to receive a continuous read out of the current status of the car's sensors which are used by the minicomputer. This proved to be one of the invaluable debugging aids of the overall system.

Concluding Comments

Projects involved with robotics are a logical extension of microcomputer technology. The possibilities of building such dedicated "artificially intelligent" machines are almost limitless. It is not unreasonable to think that personal computer experimenters could build a robotic machine at home. However, a little forethought is worth a lot of time and effort in the end. Think about what the machine is going to do, and what is necessary to accomplish this. Build the system in modules which are easy to interface to each other and also easy to debug and repair. The capabilities of the machine are only bounded by the imagination of the designer. Perhaps the ultimate goal of a robotic machine is to have it perform its designated task consistently.

We think that Tee Toddler has proved an adequate fulfillment of that goal in the limited context of a light seeking mobile device. ■