ALBERT-LUDWIGS-UNIVERSITÄT
FREIBURG
INSTITUT FÜR INFORMATIK

Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr. Hans Burkhardt

Handbook for the Use of the Labmate-Robot

Interner Bericht 6/98

X.-F. Zhang

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Lehrstuhl für Mustererkennung und Bildverarbeitung
Universitätsgelände Flugplatz, D-79085, Freiburg

Sept. 1998

Abstract

In the following passages the reconstruction of robot from its original components is presented briefly. These components include a mobile system(Labmate), a stereo-camera mounted on a pan-tilt-unit (PTU) and a sona-ranger that measures the distance with ultrasonic transducers. The different components of the robot and their connection to host computer are diagramed and shortly described. The related demo-programs are listed and some notes are made for their execution. During this reconstruction the compatibility of the serial ports of 32-bit operating system with Labmate communication channel has been solved, which is not supplied by the manufacturer TRC.

Contents

1 Construction of Robot

The Robot as a complete system is composed of a mobile base, a vision subsystem and a distance sensing subsystem. These are all controlled by a host computer. All their components are held on the different layers of alloy-wood structured frame body upon the mobile base, as illustrated in Fig. 1. Their electrical connections are showed in Fig. 2.

2 Labmate: mobile system

The mobile base, LabMate, functions as a part for the moving of the robot. It is driven by two servo-controlled wheels coupled with four casters at each corner to oppose tipping moments. By communicating with an on-board 68HC11 microcontroller via RS232 port of a host computer, the robot can work in different modes. A detail description of the hardware and commands can be referred in User's Manual [1].

we are supplied by the manufacturer TRC a demonstration program to show how the robot works under the different modes, but the program is based on 16-bit operation system, does not function under 32-bit environment system. The reason is that some port commands are different between both systems. Therefore we have developed the fundamental serial port functions under 32 bit operation system, on which the robot can go according to the commands from host computer.

The program is compiled by MSVC 5.0++ compiler under Microsoft Developer Studio, an executable one (`demo.exe`) is under the directory `d:\home\Robort\labmate\`. After it is executed and a proper port number is given, the followings are showed on the screen:

```
L A B M A T E   D e m o n s t r a t o r

 Reset Labmate           0
 Joystick Mode           1
 Go Mode                 2 When Set Velocity +/- , for-/backwards
 Absolute Turn           3 headings:(0 .. 35999)=deg*100; radius:(mm)
 Relative Turn           4 headings:(-18000..17999)=deg*100; radius:(mm)
 Point to Point Go       5 distances in mm
 Point to Point Turn Abs 6 headings and radius as in Absolutive Turn
 Point to Point Turn Rel 7 headings and radius as in Relative Turn
 Jog                     8 Jog:  Left/Right cursor keys; Home = 0
 Set Velocity    (mm/s)  9 desired standard / turn velocity
                           Increase/Decrease: Up/Down cursor keys
 Set Acceleration  (mm/s2)    10 desired standard / desired Acceleration
 Clear Wheel Positions        11
 Set WatchDog                 12 set 4.096ms to whatever expected
 Set Position                 13 set new x position and y position
```

```
Set Heading                    14 set heading to some deg*100
EMERGENCY STOP                    Hit ESC Key
Pause                          16
Resume                         17
Exit Demonstrator              18


Enter choice:
```

by choosing different numbers, the correspondent functions can be implemented. The detail functions are described in user's manual, proper parameters for each function should be given according to its definition.

The program for a specific application can be implemented following the use of function in this demo program, after it is throughly studied.

3 Stereo-Camera: vision system and drive with frame grabber

The current cameras for the Robot are two black-white CCD cameras *KP-M1* from *Hitachi Denshi, LTd.* The number of effective pixels for both are 768(H)x493(V) under system of EIA and 756(H)x581(V) under system of CCIR. Its full specifications are described in its operation manual [2].

The two KP-M1 cameras serve as the left and right camera of a stereo-camera of the Robot, they are synchronized by same external signal. When connected with a frame-grabber, this external signal comes from the sync signal of the frame-grabber, and distributed by Kramer Black Burst Generator [3].

Because of the black and white signal of the stereo-camera, a frame-grabber with two channels [4] can satisfy the requirement of the robot vision of our situation. Therefore with the frame-grabber Matrox Meteor, stereo images can be input. The Connection between these are illustrated in Fig. 2.

With both channels defined as the left and right camera video signal inputs respectively, a program (lrgrab.c) for stereo signal input is developed. By running this program the stereo images can be grabbed after adjusting focus and are saved (named as `lrgrab_l, lrgrab_r`, tiff format), these files can be processed by Software Matrox Inspector 2.0.

After studying the functions of MIL (which are fully described in software manual[5]), programs can be developed to implement different functions as required.

4 Pan-Tilt-Unit(PTU): vision orientating system

Pan-Tilt-Unit(PTU) is a computer-controlled unit for accurate positioning of certain payloads. As described in former section two cameras are mounted on the PTU and function as a stereo-camera, which serves as a vision head of the robot. By communicating with a host computer through RS232 serial port, PTU can be controlled to pan and tilt at a given angle with desired speed and acceleration. Specifications and commands are given in detail in its user's manual [6].

The current demonstration program

```
d:\home\robort\ptu\test0\test_ptu.exe
```

performs an evaluation to all specified functions, in Appendix C this process is showed.

The program

```
d:\home\robort\ptu\test1\test_ptu1.exe
```

is a modification to former and can be as an example for further development. It controls a pan movement to a given degree.

5 SonaRanger: distance sensing

Sonaranger is a set of IC boards with 8 transducers attached to measure the distance of an obstacle ahead. It is also communicated via an RS232 serial port. Its functions and commands are described in manual [7].

The driving program `d:\home\robort\sona\SONA_DEMO.bat` shows a board with 8 sensors to measure eight distances.

References

[1] LABMATE, User Manual, Version 5.21L-f, TRC, Transitions Research Corporation.

[2] KP-M1, Black and White CCD Camera, operation manual

[3] KRAMER, Broadcast 6003/B, Instruction Manual

[4] Matrox Meteor, Hardware Installation Manual

[5] MIL-Lite (Version 3.1), User Guide and Command Reference

[6] Computer Controlled PAN-TILT-UNIT, Model PTU

[7] SonaRanger^{TM}, User Manual, Release 4.Da, TRC.

A Notes on the program of Labmate

```
SOFTWARE FOR LABMATE DEMO (1)
=============================

A.Programs List:
----------------

1. Demo.c: The main program
2. Dr_56.c: Subroutines to directly invoke LABMATE commands
3. Delay.c: Subroutine for a delay
4. my.c: Subroutine attached with serial ports
5. Dr.h: Header file of Macro and Subroutines declaration
6. dr56.h: Header file of a short compensation
7. ptu.h: Header file from PAN-TILT_UNIT
8. ptu.c: from PAN-TILT_UNIT
9. W32SERIA.c: from PAN-TILT_UNIT
10. W32SERIA.h: Header file from PAN-TILT_UNIT
11. OPCODES.h: from PAN-TILT_UNIT

B. Include Relations:
---------------------

1.Demo.c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include "dr.h"
#include "d:\Home\ptu-temp\include\ptu.h"

2.Delay.c
#include<time.h>

3.Dr_56.c
#include <stdio.h>
#include "dr.h"
#include "dr56.h"

4.my.c
#include <stdio.h>
#include <ctype.h>
#include "\home\ptu-temp\include\ptu.h"

C. Arbeitsbereich "Demo" in Developer Studio
```

Delay.c
Demo.c
Dr_56.c
my.c
W32SERIA.c

Externe Abhngigkeiten:

dr.h
dr56.h
opcodes.h
ptu.h
w32seria.h

D. Some Notes:
---

1. In Demo.c the beginning is modified following Test.c of PTU software, type portstream_fd and PORT_NOT_OPENED are defined in W32SERIA.H, ptu.h can be replaced by it.
2. In dr.h the basic serial functions (file header) including SETPORT,CHRIN, CHOUT are defined
3. SETPORT, CHRIN, CHROUT are from open_host_port, GetSerialChar, SerialOut in Ptu.c of PTU software.
4. Delay.c is from the program in sona55
5. In Dr_56.c the calls to CHRIN, CHROUT are modified same format as in DR.H head file, SETPORT is at beginning of Demo.c.
6. If the Labmate connected from host is not powered on, Demo.c will loop at CHRIN (SerialByteIn) in W32SERIAL.C as is dead.
7. In void ss_out() of Dr_56.c, ack is unsigned int, while CHAR chrin_lobyte should also unsigned char, otherwise when ack = chrin_lobyte() will result in wrong ack and an unstopped loop after this assignment.
   Solution:
   (a) modified as ack = (unsigned char) chrin_lobyte();
   (b) typedef unsigned char CHAR, but this method does not always work well, there is a warning: new type has not really function.
   (dr.h(44) : warning C4142: Neudefinition eines Typs ohne Auswirkungen)
   This causes the program enter into unended loops.
   After with CTRL-C breaking, the key on the labmate should be pressed before the next run.
8. In Dr_56.c, modify "CHAR chrin_lobyte()" as "unsigned char chrin_lobyte()", the display can become integer in right range.

B Notes on the program of PTU

```
PAN-TILT-UNIT SOFTWARE

1. The programs in PTU(PAN-TILT-UNIT) software:

***********
Test.c
***********

#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>
#include "..\include\ptu.h"

***********
Ptu.h
***********

#include "..\include\W32SERIA.H" or
#include "..\include\W16SERIA.H" or
#include "..\include\DISSERIA.H" and
#include "..\include\opcodes.H"
 Procedures declaration, such as
 extern portstream open_host_port(char *)

***********
Opcodes.h
***********

Operation codes definition for PTU, such as
#define TILT_SET_REL_POS 132 etc.

***********
PTU.C
***********

#include <stdio.h>
#include <ctype.h>
#include "..\include\ptu.h

For the procedures named in ptu.h, implement them with
```

the basic procedures in W32/16/DosSERIA.h.

e.g.

```
portstream_fd open_host_port(char *portname)
{
current_host_port=openserial(portname);
return current_host_port;
}
```

****
W32/16/DosSeria.h
****

Basic procedures of serial ports for w32/16 or Dos system

e.g.
openserial(portname)

2. Structures

```
Test.c ---> Ptu.h ---> Opcodes.h
  |  \
  |   \
  |    \
Ptu.c W32/16/DosSeria.h
```

3. Application

A specific application can follow the styles of Test.c, and
use MSVC to make an EXE file.

4. Examples:

(1) W32.exe under \home\ptu-temp\w32, a program to test
 all functions of PTU.
(2) W32.exe under \home\ptu-temp\twinnt, a program to pan
 a given value corresponding to an angle.

C Execution result of the PTU program, functions evaluation

```
****** PAN-TILT BINARY TEST PROGRAM, v1.08.01
****** Serial Port Interface Driver, Win32 v1.08.09
****** (c)1997, Directed Perception, Inc. All Rights Reserved.

Enter the COM port number the PTU is attached to: 2
You selected COM2. Is this OK? (enter 'y' or 'n'): y
Serial port COM2 initialized


Controller firmware v1.7.8 is compatible


Pending commands and input buffer flushed


Firmware version:
Pan-Tilt Controller v1.07.08b,
(C)1995 Directed Perception, Inc., All Rights Reserved


Firmware version command executed successfully

RESET_PTU executed and verified

PAN_SET_ABS_POSITION executed
AWAIT_COMPLETION executed (0)

PAN_CURRENT_POSITION_QUERY issued and verified

PAN_DESIRED_POSITION_QUERY issued and verified
PAN_SET_REL_POSITION executed
AWAIT_COMPLETION executed (0)

TILT_SET_ABS_POSITION executed
AWAIT_COMPLETION executed (0)

TILT_CURRENT_POSITION_QUERY issued and verified

TILT_DESIRED_POSITION_QUERY issued and verified
TILT_SET_ABS_POSITION executed
```

```
AWAIT_COMPLETION executed (0)

DISABLE_POSITION_LIMITS executed
DISABLE_POSITION_LIMITS verified
ENABLE_POSITION_LIMITS executed
ENABLE_POSITION_LIMITS verified

PAN_HOLD_POWER_LEVEL set and query executed and verified
TILT_HOLD_POWER_LEVEL set and query executed and verified
PAN_MOVE_POWER_LEVEL set and query executed and verified
TILT_MOVE_POWER_LEVEL set and query executed and verified

PAN_CURRENT_SPEED_QUERY issued and verified
TILT_CURRENT_SPEED_QUERY issued and verified

PAN_SET_ABS_SPEED executed
PAN_DESIRED_SPEED_QUERY issued and verified
TILT_SET_ABS_SPEED executed
TILT_DESIRED_SPEED_QUERY issued and verified

PAN_SET_REL_SPEED executed
PAN_DESIRED_SPEED_QUERY issued and verified
TILT_SET_REL_SPEED executed
TILT_DESIRED_SPEED_QUERY issued and verified

PAN_MINIMUM_POSITION_QUERY RETURNED: -12369
PAN_MAXIMUM_POSITION_QUERY RETURNED: 12370
TILT_MINIMUM_POSITION_QUERY RETURNED: -3704
TILT_MAXIMUM_POSITION_QUERY RETURNED: 2469
Are the above values correct? (Enter 'y' or 'n'):
Min/Max position queries PASSED


SET_VERBOSE_ASCII_OFF executed
SET_VERBOSE_ASCII_OFF verified

SET_VERBOSE_ASCII_ON executed
SET_VERBOSE_ASCII_ON verified

DISABLE_ECHO executed
DISABLE_ECHO verified
ENABLE_ECHO executed
ENABLE_ECHO verified

PAN_SET_BASE_SPEED executed
```

```
PAN_DESIRED_BASE_QUERY issued and verified
PAN_CURRENT_BASE_QUERY issued and verified

TILT_SET_BASE_SPEED executed
TILT_DESIRED_BASE_QUERY issued and verified
TILT_CURRENT_BASE_QUERY issued and verified

PAN_SET_ACCELERATION executed
PAN_ACCELERATION_QUERY issued and verified
PAN_SET_ACCELERATION executed
PAN_ACCELERATION_QUERY issued and verified

TILT_SET_ACCELERATION executed
TILT_ACCELERATION_QUERY issued and verified
TILT_SET_ACCELERATION executed
TILT_ACCELERATION_QUERY issued and verified

PAN_SET_LOWER_SPEED_LIMIT executed
PAN_LOWER_SPEED_QUERY issued and verified
PAN_SET_LOWER_SPEED_LIMIT executed
PAN_LOWER_SPEED_QUERY issued and verified

TILT_SET_LOWER_SPEED_LIMIT executed
TILT_LOWER_SPEED_QUERY issued and verified
TILT_SET_LOWER_SPEED_LIMIT executed
TILT_LOWER_SPEED_QUERY issued and verified

PAN_SET_UPPER_SPEED_LIMIT executed
PAN_UPPER_SPEED_QUERY issued and verified
PAN_SET_UPPER_SPEED_LIMIT executed
PAN_UPPER_SPEED_QUERY issued and verified

TILT_SET_UPPER_SPEED_LIMIT executed
TILT_UPPER_SPEED_QUERY issued and verified
TILT_SET_UPPER_SPEED_LIMIT executed
TILT_UPPER_SPEED_QUERY issued and verified

PAN_RESOLUTION_QUERY RETURNED: 0.771 arc min
TILT_RESOLUTION_QUERY RETURNED: 0.771 arc min
Are the above values correct? (Enter 'y' or 'n'): y
Resolution queries PASSED


SET_IMMEDIATE_COMMAND_MODE executed
SET_IMMEDIATE_COMMAND_MODE verified
```

```
SET_SLAVED_COMMAND_MODE executed
SET_SLAVED_COMMAND_MODE verified
SET_SLAVED_COMMAND_MODE executed

HALT(ALL) executed
HALT(ALL) verified
HALT(TILT) executed
HALT(TILT) verified
HALT(PAN) executed
HALT(PAN) verified

SAVE_DEFAULTS executed
RESTORE_SAVED_DEFAULTS executed
RESTORE_SAVED_DEFAULTS verified
RESTORE_FACTORY_SETTINGS executed
RESTORE_FACTORY_SETTINGS verified


BINARY COMMAND SET TESTS FINISHED SUCCESSFULLY!!


(Enter 'c' to continue):
```

D FileNames of program list

- `readme0_lab.txt`, for Labmate
- `readme1_lab.txt`, for Labmate
- `readme0_ptu.txt`, for PTU
- `readme1_ptu.txt`, for PTU
- `readme_sona.txt`, for SonaRanger

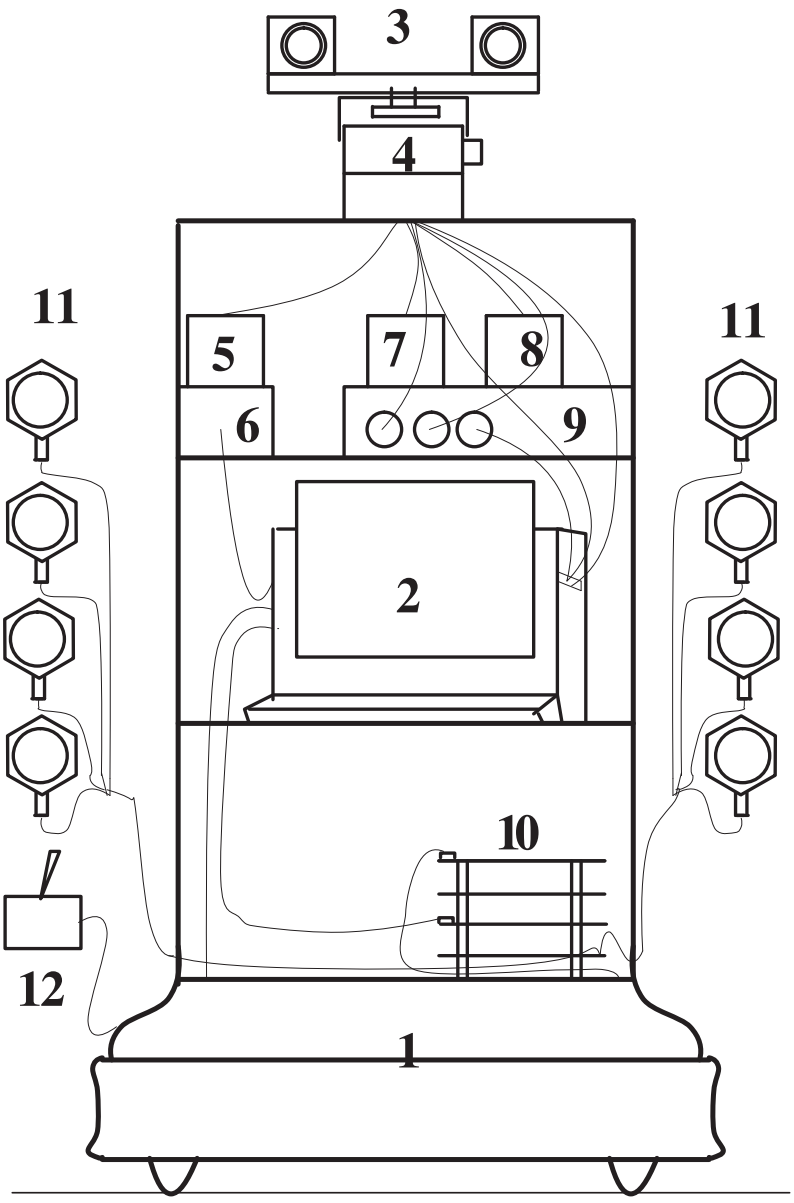

Figure 1: Construction of Robot: 1. Labmate with bumper protect; 2. Host computer; 3. Stereo-Camera; 4. Pan-Tilt-Unit (PTU); 5. Power supply for PTU; 6. PTU Controller; 7. Power supply for left camera; 8. Power supply for right camera; 9. Black burst generator; 10. SonaRanger IC boards; 11 Ultrasound transducers (up to 8) and 12. Joystick.

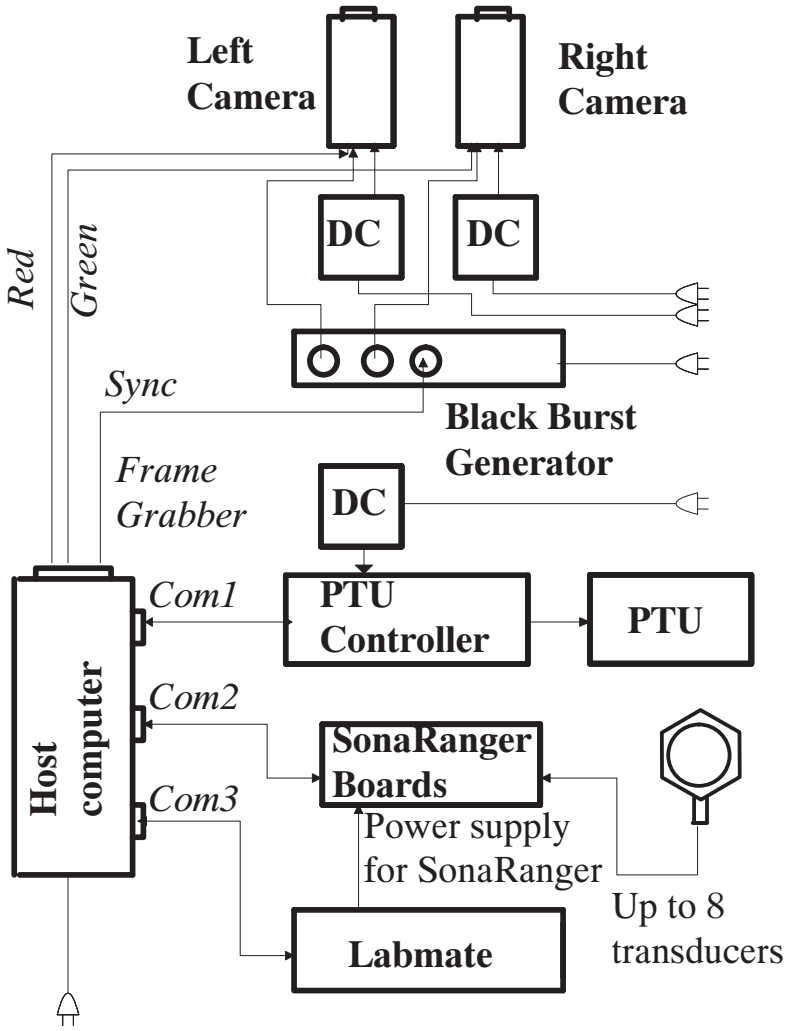

Figure 2: Connection between parts of robot.