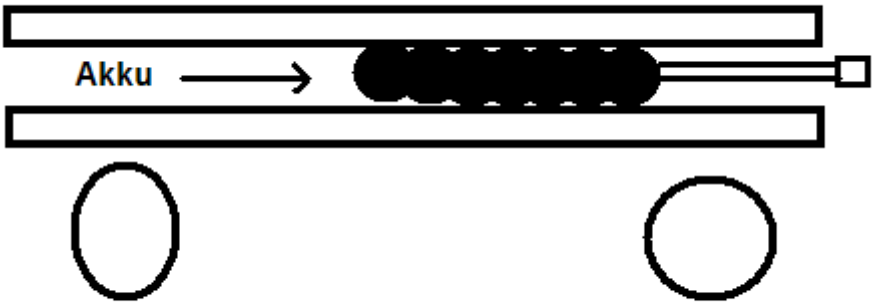
Mobiler Fussball-Roboter

Gebaut und programmiert von:
Ulrich Weger

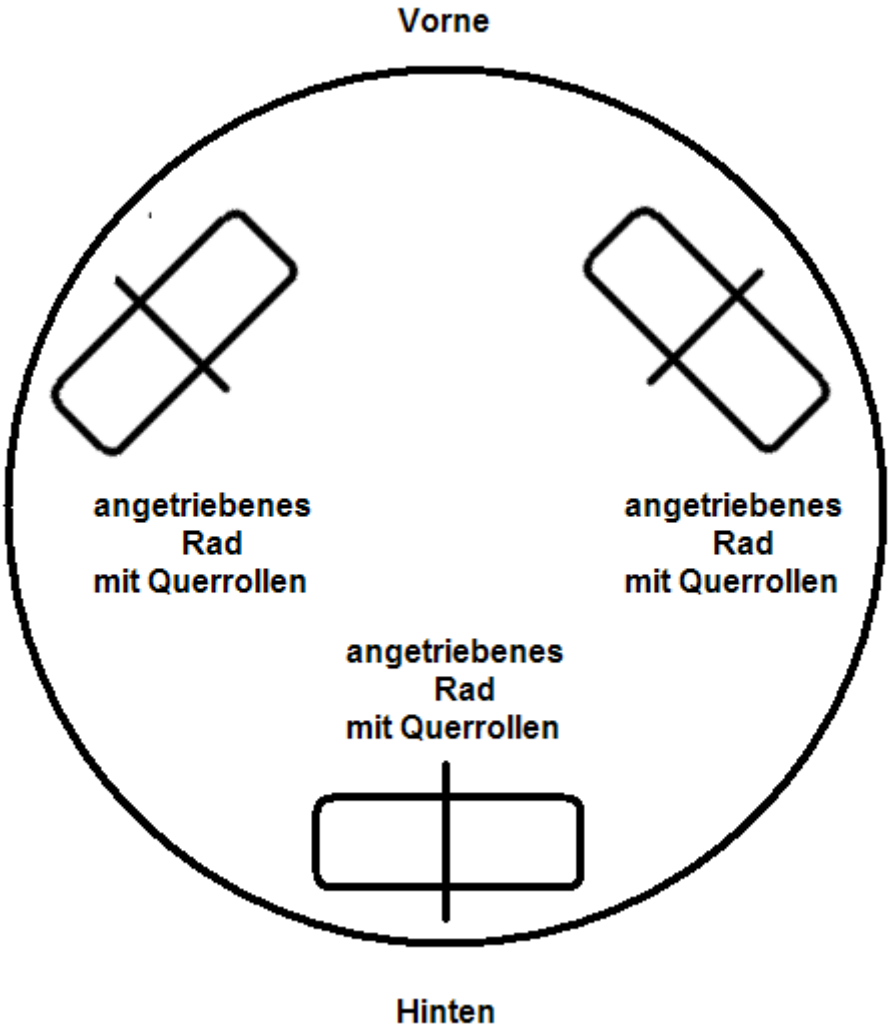
In dem Kurs "mobile Roboter" war es das Ziel, mithilfe eines Bausatzes einen omni-direktionalen Roboter zu bauen und programmieren, der Roboterfussball spielen kann. Roboterfussball ist eine Disziplin, bei der auf einer umrandeten Spielfläche ein Ball, der Imfrarotstrahlen aussendet, vom eigenen Roboter in das gegnerische Tor zu bringen ist, welches ebenfalls durch Infrarotstrahlung mit festgelegter Frequenz erkannt werden kann. Der Roboter ist mit entsprechenden Sensoren ausgestattet, die ihm das erkennen sowohl des Balls, auch als des gegnerischen Tors ermöglichen. Ferner verfügt der Roboter über Abstands-sensoren, die die Entfernung zu einem Hinderniss, wie den Wänden und anderen Robotern erkennen. Direkter Kontakt mir einem Hindernis kann über Sensoren an einem Stossstange, welche die Roboterplattform unrunden, registriert werden. Angetrieben wird der Roboter von drei Elektromororen, die über eine kleines Getriebe direkt mit jeweils einem Rad verbunden sind. Um das gegnerische Tor zu erkennen, können die vom Tor gesendetetn Infrarotstrahlen mit einem speziell konstruiertem Torsensor gemessen und so der Robot auf das Ziel ausgerichtet werden.

Das Herzstück des Robots ist eine Platine, auf der ein kleiner Computer verbaut ist, der in einem C-Dialekt programmiert werden kann. Es handelt sich dabei um das Aksen-Board, das von der FH-Brandenburg entwicklet wurde:

Das Board verfügt über analoge und digitale Eingänge, sowie über 4 Ausgänge, an denen jeweils ein Motor angeschlossen werden kann und zusätzlich noch über die Möglichkeit, Modellbau-Servos anzusteuern. Versorgt wird das Board von einem Modellbau-Akku, der zwischen den Plattformen der Roboters Platz findet:

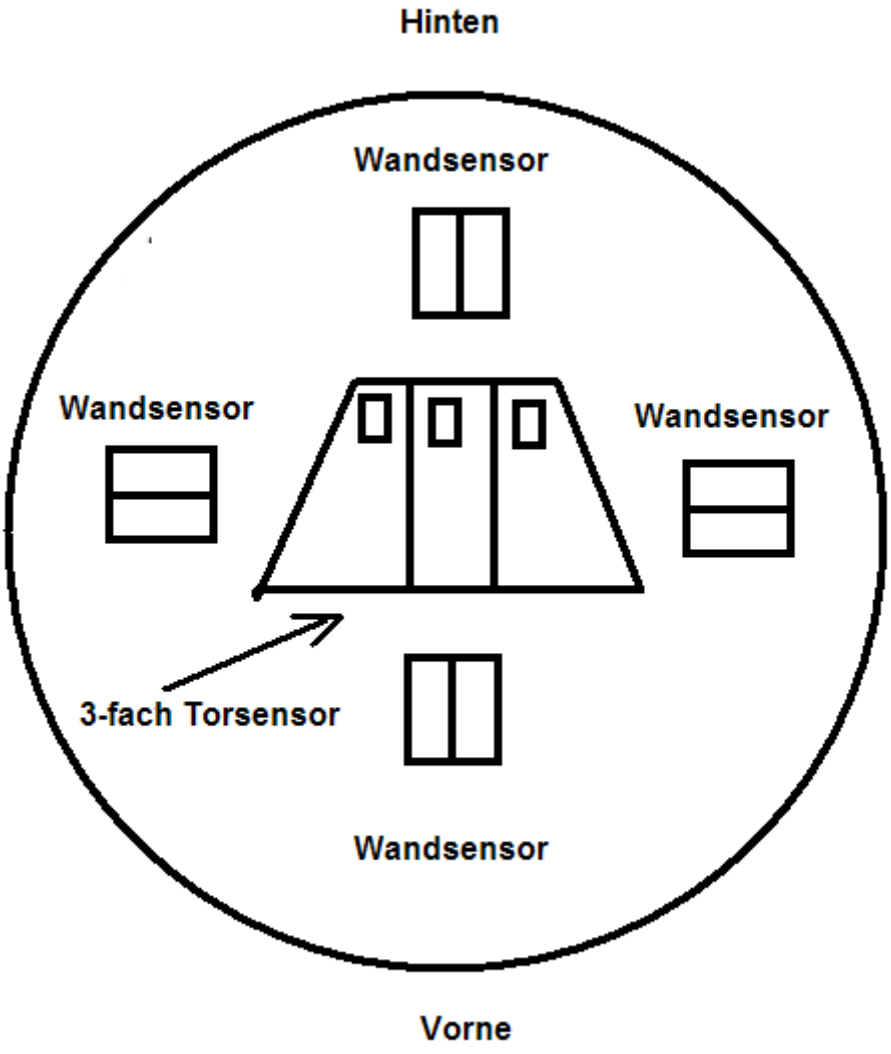


Die Omnidirektionalität des Robots wird durch die drei, jeweils 120 Grad versetzt angebrachten Räder realisiert, an deren Lauffläche wiederum 90 Grad versetzte Rollen angebracht sind, die ermöglichen, dass sich der Robot auch senkrecht zur Laufrichtung eines Rades bewegen kann:

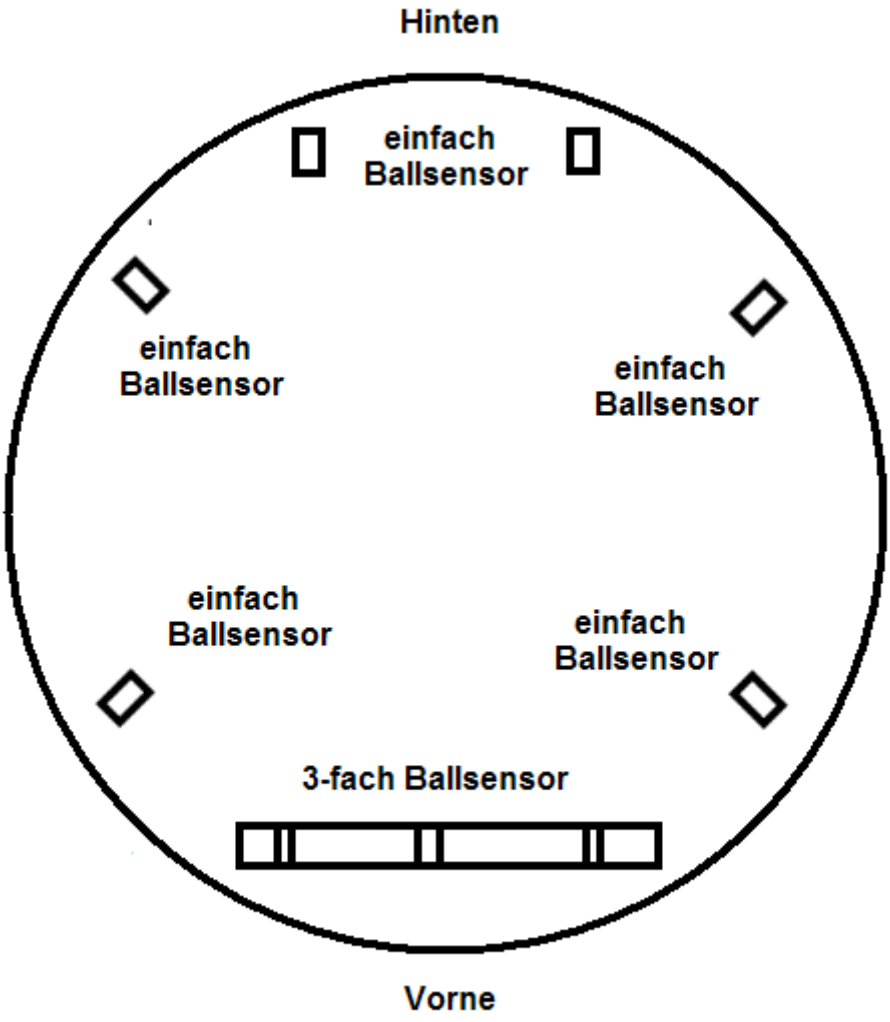


Die umliegende Wände un andere Hindernisse erkennt der Robot durch die vier rundherum angebrachten Wandsensoren, die einen dem Abstand entsprechenden Spannungswert an jeweils einen Analogeingang des Aksenboards anlegen lassen.

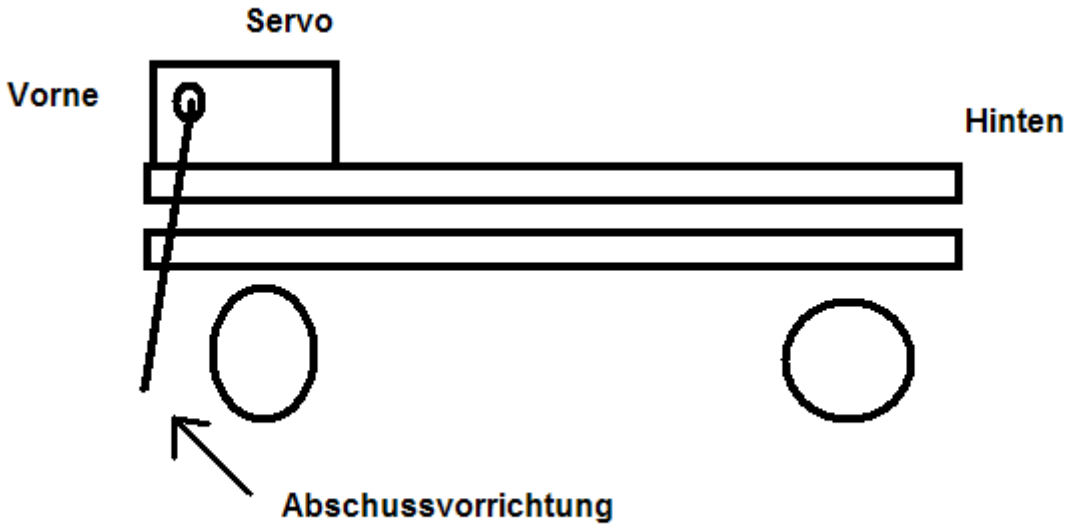
Ebenfalls auf der Skizze zu sehen ist die Konstruktion des Torsensors, der aus drei einzelnen Sensoren besteht, die in einem speziellen Gehäuse verbaut sind. Durch die Unterschiede im Messwert zwischen dem rechten und linken Sensor kann die Bewegungsrichtung zum Tor ermittelt werden. Auf den mittleren Sensor kann nur Strahlung treffen, wenn der Winkel zum Ziel sehr klein ist:



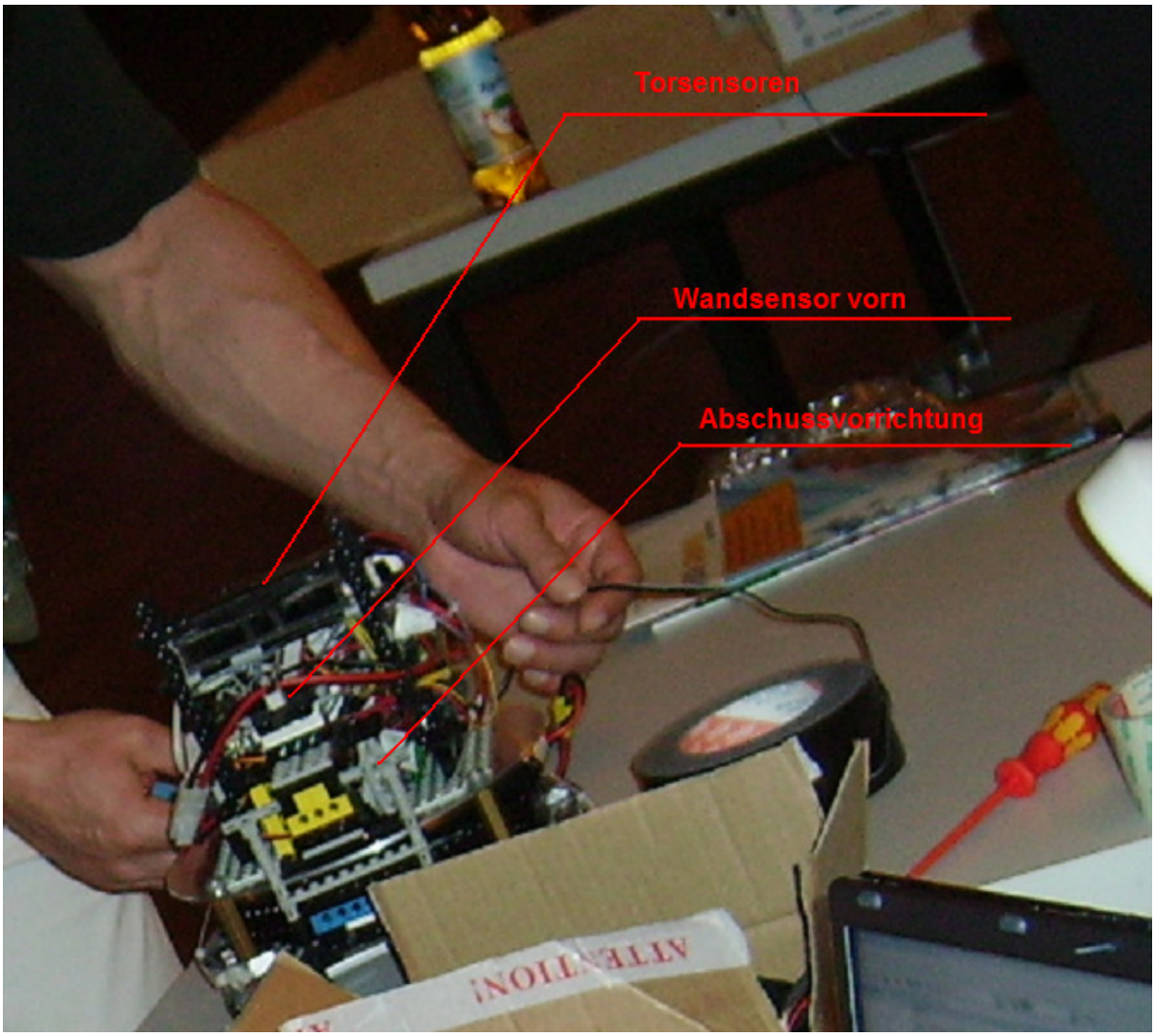
9 Ballsensoren sind um die Plattform verteilt angebracht, die geben jeweils einen analogen Messwert an jeweils einen Eingang des Boards. Vorne sind drei Ballsensoren zusammen verbaut worden, sie sollen den Ball auch auf grössere Entfernung erkennen:

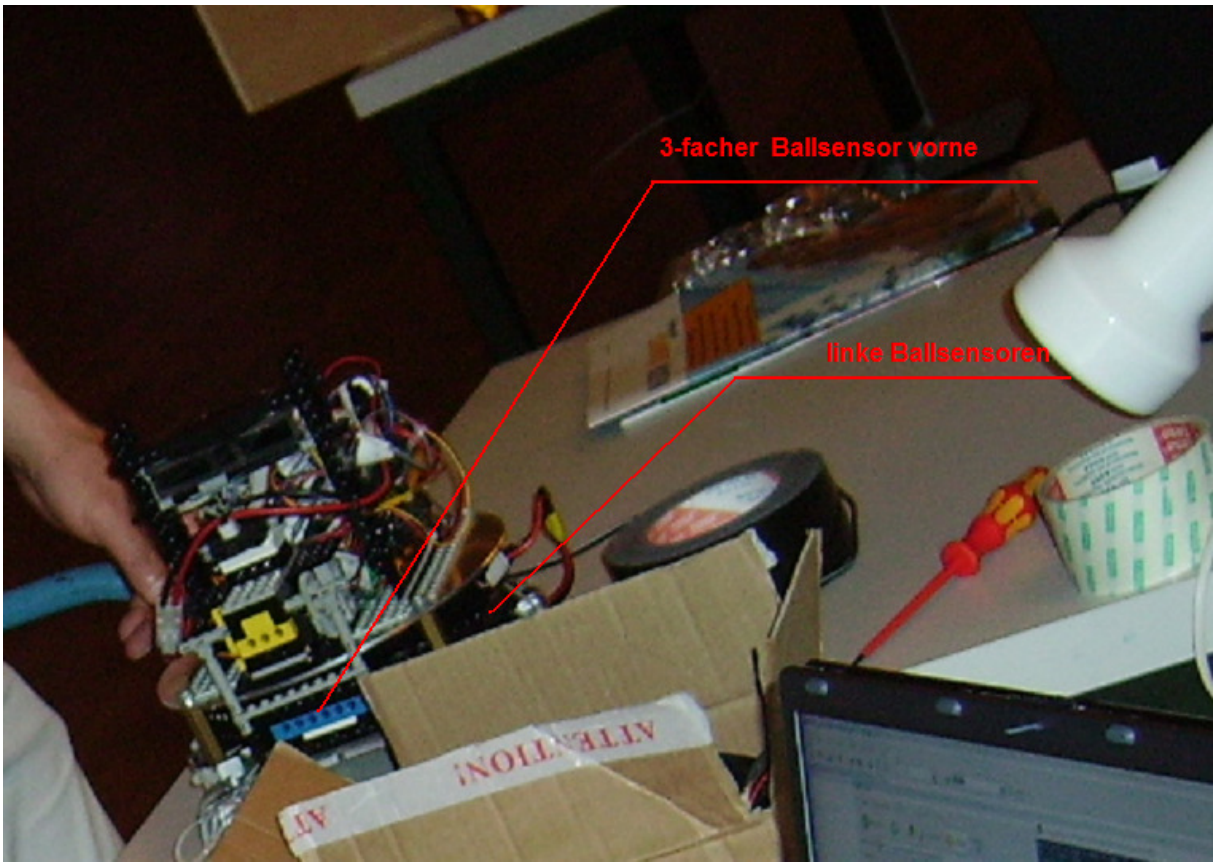


Vorne befindet sich eine Abschussvorrichtung, die aus einem Servo besteht, an dessen Achse zwei Lego-Stangen angebracht worden:



Hier zwei Fotos von dem Robot:





Der Quellcode, zuerst das Header-File:

```
/*

Dies ist die endgültige Implementierung zum Roboter-Fussball, Headerfile

von ulrich weger

*/

#define ON 1
#define OFF 0

#define VOR 0
#define RUECK 1

#define MOTOR_0      0
#define MOTOR_1      1
#define MOTOR_2      2
#define MOTOR_3      3

#define MO_STOP      0
#define VERYSLOW     3
#define SLOW         4
#define MIDDLE         6
#define FAST         8
#define VERYFAST    10

#define TRUE         1

#define FZ_LONG    0//400
#define FZ_SHORT   100//200
//#define FZ_SHORT   150
#define VERYLONG   250
#define WAND        150
#define TORGEFUNDEN   0
#define IR_PERIODEN_OK 3 // !!!!!

#define FALSE 0
#define TRUE 1

#define FREQ1 4 /* 4=125 Hz, 5=100 Hz */
#define FREQ2 5 /* 4=125 Hz, 5=100 Hz */

#define BS_EMPF  100
#define BS_EMPF_TOR  80
//#define BS_EMPF_TOR  70

#define BS_EMPF_M 200
#define BS_EMPF1 200

#define BS_EMPF_L 100
#define BS_EMPF_1 200
#define BS_EMPF_2 200
#define BS_EMPF_3 200

#define WS_EMPF 140// Empfindlichkeit der Wandsensoren
#define WS_EMPF_V 180// Empfindlichkeit des vorderen Wandsensors

#define WS_EMPF_TOR 190//120

void sucheTor(void);
```

Und das C-File:

```
/*

Dies ist die endgültige Implementierung zum Roboter-Fussball

von ulrich weger

*/

//dips:
//0: tor-frequenz
//1:  ball in mitte - anstoss mit start vom tor
//2: motortest + kicktest
//3: debug meldung mit verzoegerung


#include <stdio.h>
#include <stub.h>
#include "nr5_new.h"


#define STOP {\
            motor_pwm(MOTOR_0, MO_STOP);\
            motor_pwm(MOTOR_1, MO_STOP);\
            motor_pwm(MOTOR_2, MO_STOP);\
            }

#define VORWAERTS {\
            motor_richtung(0,RUECK);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, MO_STOP);\
            motor_pwm(MOTOR_2, FAST); \
            sleep(FZ_LONG);\
            }

#define POWERVORWAERTS {\
            motor_richtung(0,RUECK);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, MO_STOP);\
            motor_pwm(MOTOR_2, VERYFAST); \
            sleep(FZ_LONG);\
            }


#define RUECKWAERTS {\
            motor_richtung(0,VOR);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, MO_STOP);\
            motor_pwm(MOTOR_2, VERYFAST);\
            sleep(FZ_LONG);\
            }

#define DREHUNG_RECHTS_VERYFAST {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, VERYFAST);\
            }

#define DREHUNG_RECHTS {\
```

```
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(FZ_SHORT);\
            }

#define DREHUNG_LINKS {\
            motor_richtung(0,VOR);\
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(FZ_SHORT);\
            }

#define DREHUNG_RECHTS_1 {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(150);\
            }

#define DREHUNG_LINKS_1 {\
            motor_richtung(0,VOR);\
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(150);\
            }

#define DREHUNG_RECHTS_2 {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(250);\
            }

#define DREHUNG_LINKS_2 {\
            motor_richtung(0,VOR);\
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(250);\
            }

#define DREHUNG_RECHTS_3 {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(400);\
            }

#define DREHUNG_LINKS_3 {\
            motor_richtung(0,VOR);\
```

```
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, FAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(400);\
            }

#define DREHUNG_RECHTS_SLOW {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, MIDDLE);\
            motor_pwm(MOTOR_1, MIDDLE);\
            motor_pwm(MOTOR_2, MIDDLE);\
            sleep(FZ_SHORT);\
            }

#define DREHUNG_LINKS_SLOW {\
            motor_richtung(0,VOR);\
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, MIDDLE);\
            motor_pwm(MOTOR_1, MIDDLE);\
            motor_pwm(MOTOR_2, MIDDLE);\
            sleep(FZ_SHORT);\
            }

//rueckzugs-bewegung
#define BEWEGUNG_LINKS {\
            motor_richtung(0,VOR);\
            motor_richtung(1,RUECK);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, MO_STOP);\
            sleep(FZ_LONG);\
            }

//rueckzugs-bewegung
#define BEWEGUNG_RECHTS {\
            motor_richtung(1,VOR);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, MO_STOP);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, VERYFAST);\
            sleep(FZ_LONG);\
            }

//rueckzugs-bewegung mit drehung
#define RUECKWAERTS_DREHUNG_RECHTS {\
            motor_richtung(0,VOR);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, FAST);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, MIDDLE);\
            sleep(FZ_LONG);\
            }

//rueckzugs-bewegung mit drehung
#define RUECKWAERTS_DREHUNG_LINKS {\
            motor_richtung(0,VOR);\
            motor_richtung(1,VOR);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, MIDDLE);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, FAST);\
            sleep(FZ_LONG);\
            }

#define VORWAERTS_DREHUNG_RECHTS {\
```

```
            motor_richtung(0,RUECK);\
            motor_richtung(1,VOR);\
            motor_richtung(2,RUECK);\
            motor_pwm(MOTOR_0, MIDDLE);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, VERYFAST);\
            sleep(FZ_LONG);\
            }

#define VORWAERTS_DREHUNG_LINKS {\
            motor_richtung(0,VOR);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, MIDDLE);\
            sleep(FZ_LONG);\
            }

#define TANZE_RECHTS {\
            motor_richtung(1,RUECK);\
            motor_pwm(MOTOR_0, MO_STOP);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, MO_STOP);\
            sleep(FZ_SHORT);\
            }

#define TANZE_LINKS {\
            motor_richtung(1,VOR);\
            motor_pwm(MOTOR_0, MO_STOP);\
            motor_pwm(MOTOR_1, VERYFAST);\
            motor_pwm(MOTOR_2, MO_STOP);\
            sleep(FZ_SHORT);\
            }

#define SCHIEBE_RECHTS {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,RUECK);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, MIDDLE);\
            motor_pwm(MOTOR_2, VERYFAST);\
            sleep(FZ_SHORT);\
            }

#define SCHIEBE_LINKS {\
            motor_richtung(0,RUECK);\
            motor_richtung(1,VOR);\
            motor_richtung(2,VOR);\
            motor_pwm(MOTOR_0, VERYFAST);\
            motor_pwm(MOTOR_1, MIDDLE);\
            motor_pwm(MOTOR_2, VERYFAST);\
            sleep(FZ_SHORT);\
            }

#define KICK {\
            servo_arc(0, 100);\
            servo_arc(0, 80);\
            sleep(200);\
            servo_arc(0, 100);\
            sleep(200);\
            }

#define LED0_AUS led(0,0);
#define LED0_AN led(0,1);

#define LED1_AUS led(1,0);
#define LED1_AN led(1,1);

#define LED2_AUS led(2,0);
#define LED2_AN led(2,1);
```

```
#define LED3_AUS led(3,0);
#define LED3_AN led(3,1);



//Ballsensoren
#define bsml_P (analog(0))//mitte
#define bsmr_P (analog(1))//mitte
#define bshl_P (analog(2))//ganz vorne
#define bshr_P (analog(3))//ganz vorne

#define bsl_P (analog(5))//hinten
#define bslei_P (analog(6))//front
#define bsr_P (analog(7))//hinten
#define bso_P (analog(14))//nicht vorhanden

//Wandsensoren
#define wsv_P (analog(8))//ok, aber
#define wsr_P (analog(9))//evtl
#define wsl_P (analog(10))//stecker
#define wsh_P (analog(11))//prüfen

//Torsensoren
#define ts_links_P  (mod_ir0_status())//digi 04
#define ts_rechts_P (mod_ir1_status())//digi 05
#define ts_mitte_P  (mod_ir2_status())//digi 06

//Bumper
#define bumper1_P (digital_in(8))
#define bumper2_P (digital_in(9))



// Die Default_Drehrichtng, wenn keine Richtung angegeben werden kann.

unsigned int set_freq = FREQ1;

int s;
int ss;

unsigned char bsml;
unsigned char bsmr;
unsigned char bshl;
unsigned char bshr;

unsigned char bsl;
unsigned char bslei;
unsigned char bsr;
unsigned char bso;

//Wandsensoren
unsigned char wsv;
unsigned char wsr;
unsigned char wsl;
unsigned char wsh;

unsigned char ts_links,
              ts_mitte,
              ts_rechts;

unsigned int i;
unsigned int counter;
//int bs_wert;
int random;
char start;
char debug = 0;

char turn_mode = 0;

void display(char* message)
```

```
      {
      lcd_cls();
      lcd_home();
      lcd_puts(message);
      //sleep(500); //sleep verzoegerung faelscht verhalten ab
      }


void AksenMain(void){

while(dip_pin(2))
      {
      display("motortest + kick");
      DREHUNG_LINKS
      sleep(1000);
      KICK
      sleep(1000);
      }

if(dip_pin(3))
      {
      debug = 1;
      }

random = TRUE;
//int counter = 0;
  if(dip_pin(0)){
    set_freq = FREQ1;//125 Hz
    led(3,0);
    }
  else {
    set_freq = FREQ2;//100 Hz
    led(3,1);
    }


    wsv = wsv_P;
    wsh = wsh_P;

  if(wsv > (wsh + 50)){
    //start = 'd'; //drehen und fahren
      if(debug) display("start drehen und fahren");
    DREHUNG_RECHTS_VERYFAST
    sleep(1050);
    }

  bslei = bslei_P;

  if(dip_pin(1)){
      if(debug) display("start dip1 anstoss");
      POWERVORWAERTS
      sleep(7000);
      if (bslei = BS_EMPF)
            sucheTor();
    }
      POWERVORWAERTS

  while(1){

  //Ballsensoren
      bsml = bsml_P;
      bsmr = bsmr_P;
      bshl = bshl_P;
      bshr = bshr_P;

      bsl = bsl_P;
      bslei = bslei_P;
      bsr = bsr_P;
      bso = bso_P;

//Wandsensoren
```

```
      wsv = wsv_P;
      wsr = wsr_P;
      wsl = wsl_P;
      wsh = wsh_P;

      mod_ir0_takt(set_freq);
      mod_ir0_maxfehler(3);
      mod_ir1_takt(set_freq);
      mod_ir1_maxfehler(3);
      mod_ir2_takt(set_freq);
      mod_ir2_maxfehler(3);


      ts_links = ts_links_P;
      ts_mitte = ts_mitte_P;
      ts_rechts = ts_rechts_P;

//Torsensoren



      if (s == FALSE){
            led(3,0);
            s=TRUE;
            }
      else {
            led(3,1);
            s=FALSE;
            }

      if((bslei < BS_EMPF) && (ts_mitte >= IR_PERIODEN_OK) && !(wsv >
(WS_EMPF_V))){
            //habe ball + habe torrichtung + stehe nicht vor wand
            if(debug) display("aufs Tor fahren");
            POWERVORWAERTS// aufs Tor fahren
            KICK
      } else if((bslei < BS_EMPF) && ((ts_mitte >= IR_PERIODEN_OK)
                        ) &&  (wsv > (WS_EMPF_V))){
      //habe ball + habe torrichtung + stehe vor wand
      if(debug) display("zuruecksetzen");

            for(i = 1000; i>0; i--){

               if (wsh > WS_EMPF){
                  STOP
               } else {
                  RUECKWAERTS
               }
            }
      } else if(bslei < BS_EMPF_L){
            if(debug) display("suche tor");
            sucheTor();
      } else if(bshl < BS_EMPF_1){
            if(debug) display("stueck zurueck");
            RUECKWAERTS
            sleep(750);
            if(debug) display("drehung links 1");
            DREHUNG_LINKS_1
      } else if(bshr < BS_EMPF_1){
            if(debug) display("stueck zurueck");
            RUECKWAERTS
            sleep(750);
            if(debug) display("drehung rechts 1");
            DREHUNG_RECHTS_1
      } else if(bsml < BS_EMPF_2){
            if(debug) display("drehung links 2");
            DREHUNG_LINKS_2
      } else if(bsmr < BS_EMPF_2){
            if(debug) display("drehung rechts 2");
            DREHUNG_RECHTS_2
      } else if(bsl < BS_EMPF_3){
```

```
            if(debug) display("bewegung links 3");
            //BEWEGUNG_LINKS
            DREHUNG_LINKS_3
      } else if(bsr < BS_EMPF_3){
            if(debug) display("bewegung rechts 3");
            //BEWEGUNG_RECHTS
            DREHUNG_RECHTS_3
      } else if ((wsv > WS_EMPF_V) && (wsl > WS_EMPF) && (wsr > WS_EMPF)){
            if(debug) display("ueberall wand");
      //Überall ist eine Wand
            DREHUNG_LINKS
      } else if ((wsl > WS_EMPF) && (wsr > WS_EMPF)){
            if(debug) display("l+r wand");
      //links und rechts ist eine Wand
            RUECKWAERTS
      } else if((wsv > WS_EMPF_V) && (wsl > WS_EMPF)){
            if(debug) display("v+l wand");
      // vorne und links ist eine Wand
            DREHUNG_RECHTS
      } else if((wsv > WS_EMPF_V) && (wsl > WS_EMPF)){
            if(debug) display("v+r wand");
      // vorne und rechts ist eine Wand
            DREHUNG_LINKS
      } else if (wsv > WS_EMPF_V){// Vorne ist eine Wand
            if(debug) display("v wand");
            RUECKWAERTS_DREHUNG_RECHTS
            sleep(500);
            VORWAERTS_DREHUNG_RECHTS
            sleep(500);

      } else if (wsl > WS_EMPF){    //links ist eine Wand
            if(debug) display("l wand");
            VORWAERTS_DREHUNG_RECHTS
            random = TRUE;
      } else if (wsr > WS_EMPF){    //rechts ist eine Wand
            if(debug) display("r wand");
            VORWAERTS_DREHUNG_LINKS
            random = FALSE;
      } else if (wsh > WS_EMPF){
            if(debug) display("h wand");
      //hinten ist eine Wand
            POWERVORWAERTS
      } else {
            if(debug) display("keine wand");
            POWERVORWAERTS
      }
  }
}


void sucheTor(void){

      //ball richtig einfangen
      POWERVORWAERTS
      sleep(1000);



      mod_ir0_takt(set_freq);
      mod_ir0_maxfehler(3);
      mod_ir1_takt(set_freq);
      mod_ir1_maxfehler(3);
      mod_ir2_takt(set_freq);
      mod_ir2_maxfehler(3);

      for (counter = 5000;counter > 0; counter--){
            ts_links = ts_links_P;
            ts_rechts = ts_rechts_P;
            ts_mitte = ts_mitte_P;
            wsv = wsv_P;
            wsr = wsr_P;
```

```
            wsl = wsl_P;
            bslei = bslei_P;


            if(!(bslei < BS_EMPF_TOR))
                  {
                  counter = 0;
                  break;
                  }

            if((ts_mitte >= IR_PERIODEN_OK) ){
            if(debug) display("aufs tor fahren");
                  POWERVORWAERTS// aufs Tor fahren
                  KICK
                  break;
            } else if((ts_rechts == IR_PERIODEN_OK)){
                  if(debug) display("tanze rechts");
                  TANZE_RECHTS// aufs Tor fahren
                  //KICKOFF
            } else if((ts_links == IR_PERIODEN_OK)){
                  if(debug) display("tanze links");
                  TANZE_LINKS// aufs Tor fahren
            } else if (wsv > WS_EMPF_V){// Vorne ist eine Wand
                  if(debug) display("vorne wand");
                  if (wsl > WS_EMPF) turn_mode = 0;
                  if (wsr > WS_EMPF) turn_mode = 1;

                  if(turn_mode)
                        {
                        DREHUNG_LINKS
                        turn_mode = 0;
                        }
                  else
                        {
                        DREHUNG_RECHTS
                        turn_mode = 1;
                        }
                  sleep(500);

            } else if (wsr > WS_EMPF_TOR){      //rechts ist eine Wand
                  if(debug) display("rechts wand");
                  VORWAERTS_DREHUNG_LINKS
                  random = FALSE;
                  sleep(200);
            } else if (wsl > WS_EMPF_TOR){      //links ist eine Wand
                  if(debug) display("links wand");
                  VORWAERTS_DREHUNG_RECHTS
                  random = TRUE;
                  sleep(200);
            } else {
                  if(debug) display("suche torrichtung");
                  if (wsl > WS_EMPF) turn_mode = 0;
                  if (wsr > WS_EMPF) turn_mode = 1;

                  if(turn_mode)
                        {
                        SCHIEBE_RECHTS//tor suchen
                        }
                  else
                        {
                        SCHIEBE_LINKS//tor suchen
                        }
                  sleep(300);
      } //for  end
}
```

Abschliessende Bemerkungen:

Ich musste leider die Erfahrung machen, dass sich die Bedingungen bei der Vorstellung der Robots so sehr von den Bedingungen im Labor unterschieden haben, dass es erhebliche Probleme bei der Präsentationm gab. Als Haupzgrund sei hier das Licht angeführt, dass bei der Präsentation in dem anderen Gebäude von Scheinwerfern kam, die dummerweise einen sehr hohem Infrarotanteil hatten, nicht zu vergleichen mit den Leuchtstoffröhren im Labor. So kam es, dass der Robot überall Ballsignale empfing und entsprechend reagierte.

Die anderen Teams hatten zu Anfang ä#hnliche Probleme, konnten jedoch ihre Robots noch fixen, da sie Laptops mit dem Quellcode und dem Flashprogramm dabei hatten. Ich war leider in der Hinsicht nicht so gut ausgestattet und hatte mich auf die Versuche im Labor verlassen. Eine Lektion , die man daraus lernen kann ist, bei Präsentationen immer mit unerwarteten Komplikationen zu rechnen und sich möglichst auf alles versuche vorzubereiten.

Ulrich Weger