Mobile AD-Hoc Network Projekt

The Wolf Pack

Final Presentation

Bastian Görstner

Jürgen Werner

Idea of Wolf Pack

Backround from the nature:

Wolf Pack

- Consists of one Alpha Wolf
- All other follow the alpha animal

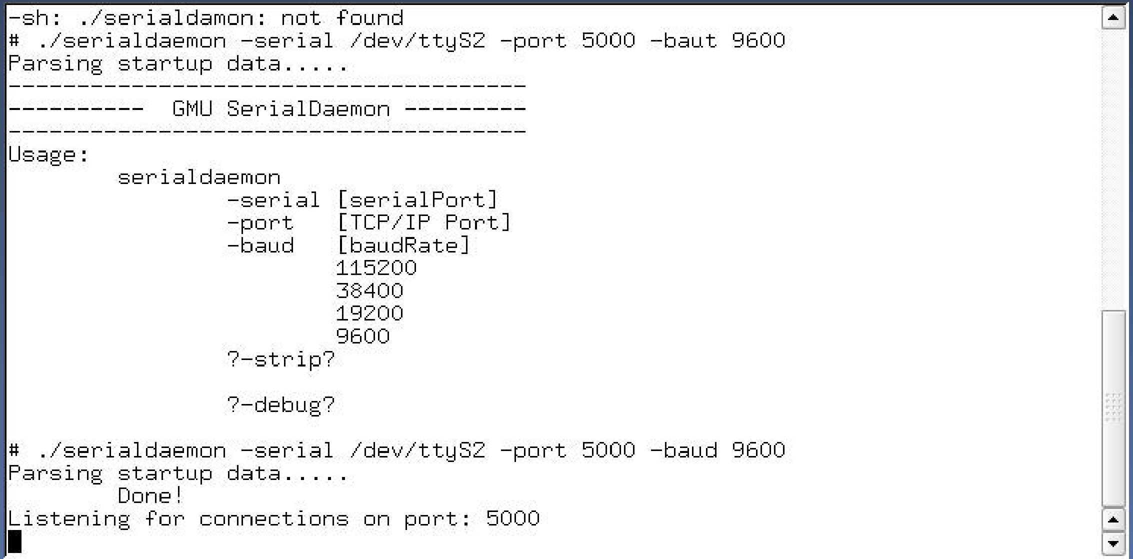
Motivation

sector of the economy:
Cheaper transportation of goods through land, air and water

military use:
safe of money and human life

Realization

- learn to work with Linux
- study the Python program language
- Gumstix connect via Mobile Ad Hoc Network
- Leader resolve his own position via GPS, this position must be transferred to all others in range
- realized with port forwarding of the GPS Serial Port (ttyS2) to the TCP Port 5000
- we used serialdaemon tool, that we have cross-compiled for the gumstix to forward



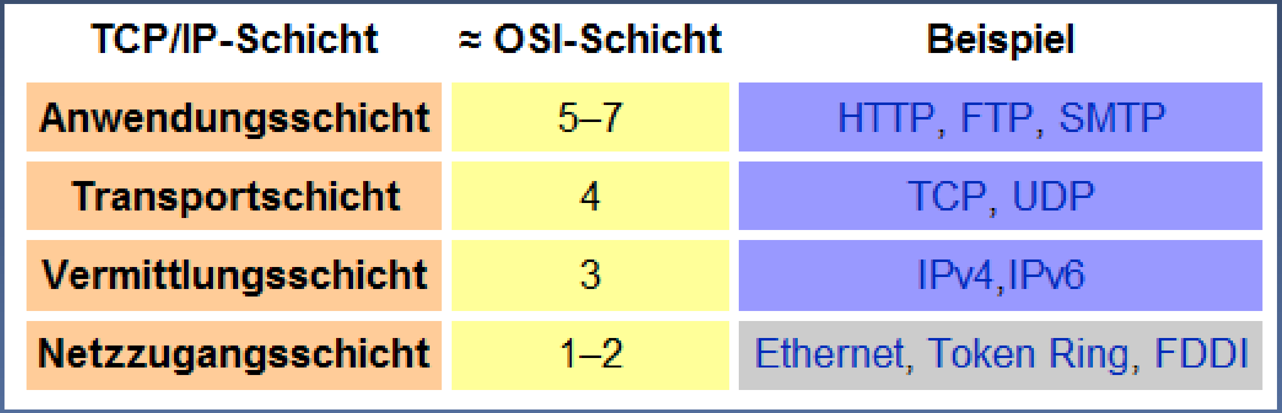
```
-sh: ./serialdamon: not found
# ./serialdaemon -serial /dev/ttyS2 -port 5000 -baut 9600
Parsing startup data.....
--------------------------------------
----------  GMU SerialDaemon ---------
--------------------------------------
Usage:
        serialdaemon
                -serial [serialPort]
                -port   [TCP/IP Port]
                -baud   [baudRate]
                        115200
                        38400
                        19200
                        9600
                ?-strip?

                ?-debug?

# ./serialdaemon -serial /dev/ttyS2 -port 5000 -baud 9600
Parsing startup data.....
        Done!
Listening for connections on port: 5000
```

Realization

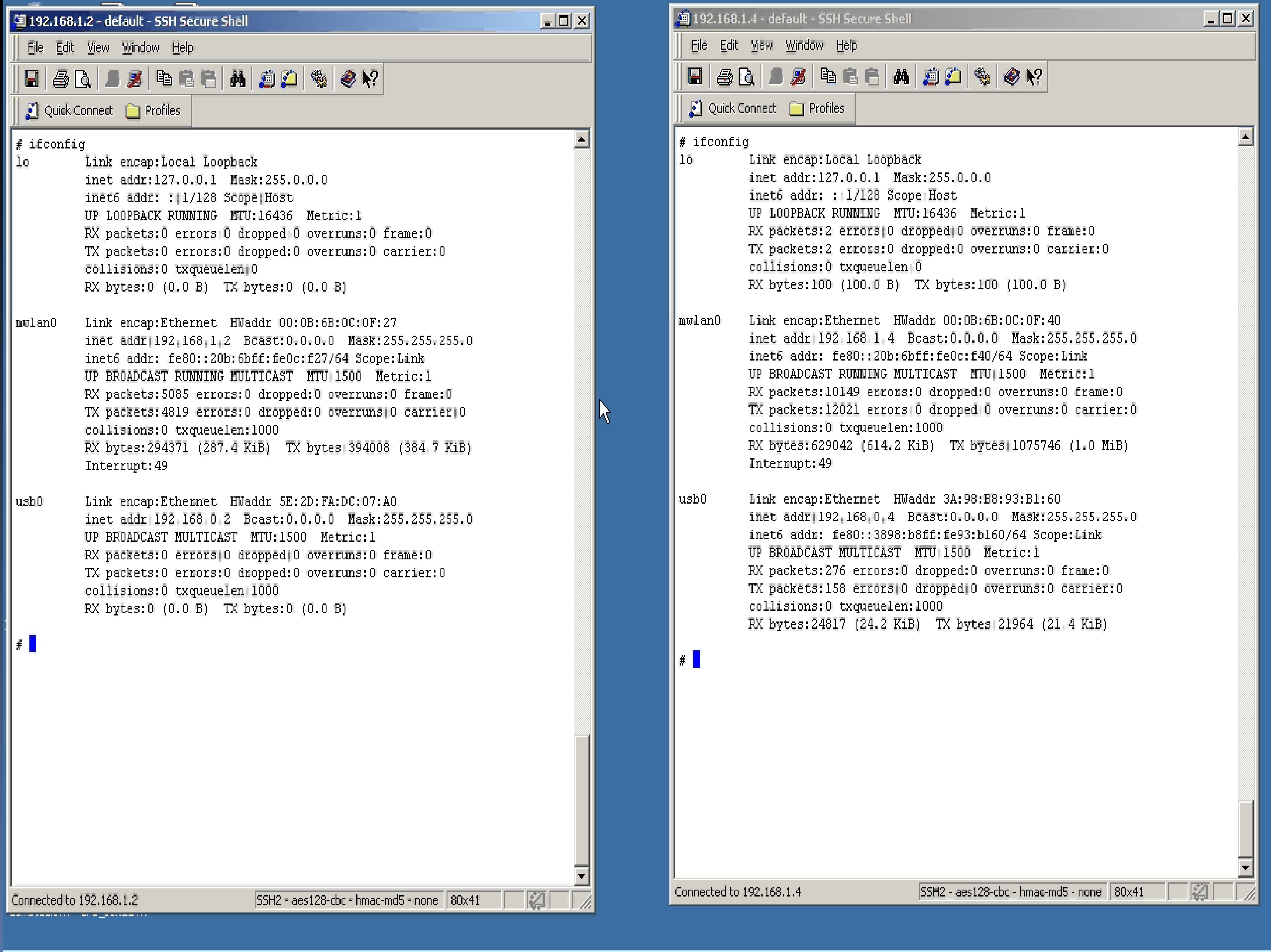
- use of Telnet on the clients to receive the GPS Signals from Host

Why we use Telnet:

- simple to install and to use
- uses TCP (Layer 4 of the OSI Model)
- disadvantage: no encryption



TCP/IP-Schicht	≈ OSI-Schicht	Beispiel
Anwendungsschicht	5–7	HTTP, FTP, SMTP
Transportschicht	4	TCP, UDP
Vermittlungsschicht	3	IPv4,IPv6
Netzzugangsschicht	1–2	Ethernet, Token Ring, FDDI


192.168.1.2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect
Profiles
Connected to 192.168.1.2
SSH2 - aes128-cbc - hmac-md5 - none
80x41

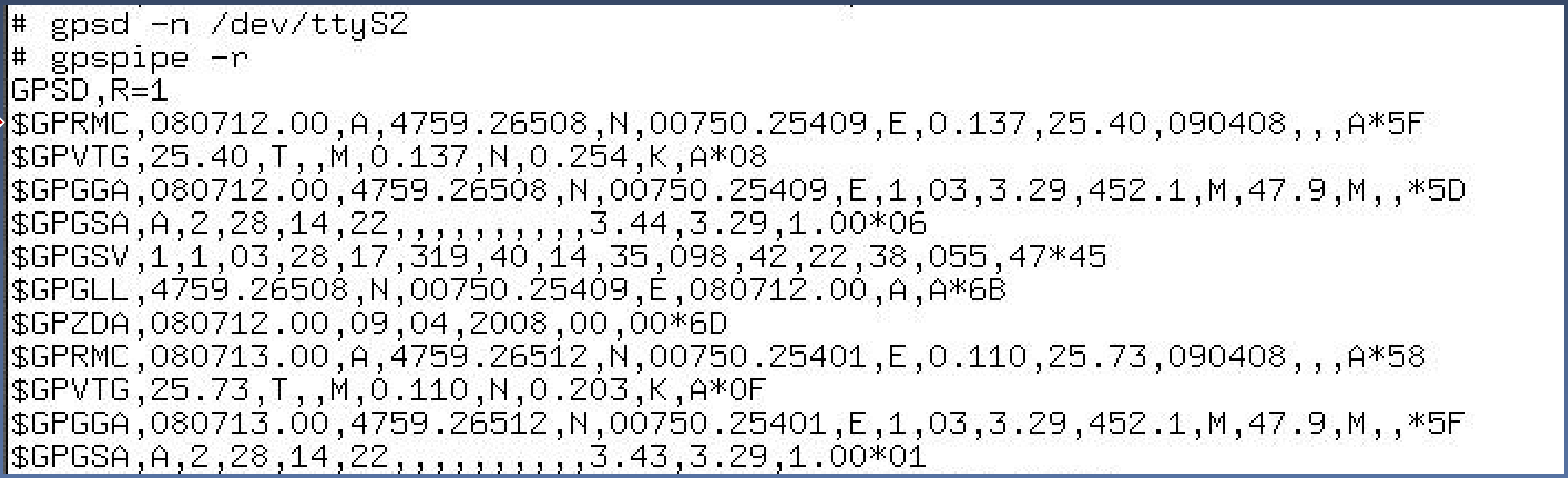
```
# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mwlan0    Link encap:Ethernet  HWaddr 00:0B:6B:0C:0F:27
          inet addr:192.168.1.2  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::20b:6bff:fe0c:f27/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5085 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4819 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:294371 (287.4 KiB)  TX bytes:394008 (384.7 KiB)
          Interrupt:49

usb0      Link encap:Ethernet  HWaddr 5E:2D:FA:DC:07:A0
          inet addr:192.168.0.2  Bcast:0.0.0.0  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

#
```

192.168.1.4 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect
Profiles
Connected to 192.168.1.4
SSH2 - aes128-cbc - hmac-md5 - none
80x41

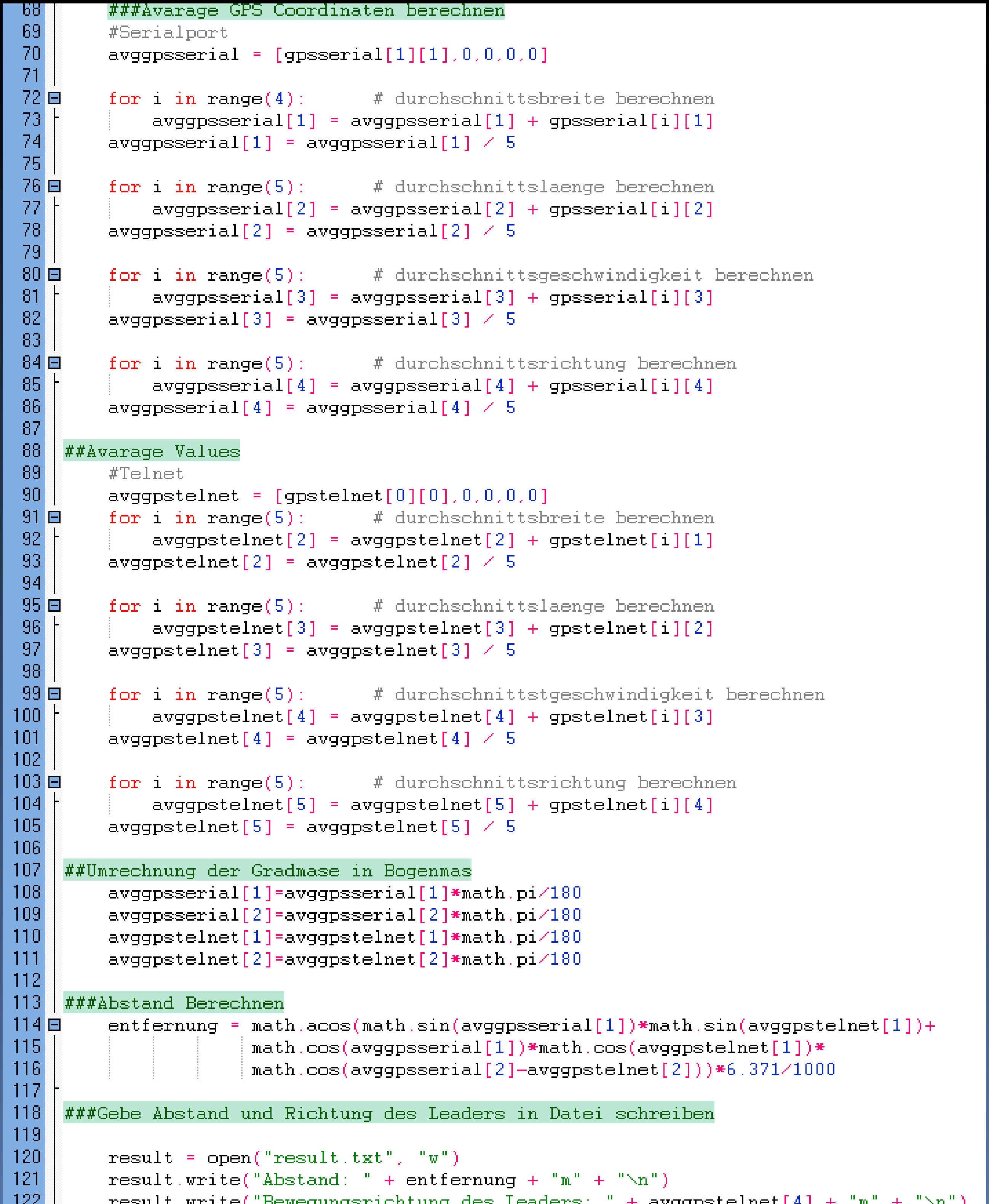
```
# ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)

mwlan0    Link encap:Ethernet  HWaddr 00:0B:6B:0C:0F:40
          inet addr:192.168.1.4  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::20b:6bff:fe0c:f40/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10149 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12021 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:629042 (614.2 KiB)  TX bytes:1075746 (1.0 MiB)
          Interrupt:49

usb0      Link encap:Ethernet  HWaddr 3A:98:B8:93:B1:60
          inet addr:192.168.0.4  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::3898:b8ff:fe93:b160/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:158 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24817 (24.2 KiB)  TX bytes:21964 (21.4 KiB)

#
```

Calculation

- write a skript in Phyton program language
- recive the GPS streams from Serial and TCP
- parse the GPRMC informations out of the streams
- calculate the average values out of 5
- calculate the distance between the two GPS positions
- write the distance and movement direction of the leader to a file
- loop the programm until the distance is under a fixed value



```
#  gpsd -n /dev/ttyS2
#  gpspipe -r
GPSD,R=1
$GPRMC,080712.00,A,4759.26508,N,00750.25409,E,0.137,25.40,090408,,,A*5F
$GPVTG,25.40,T,,M,0.137,N,0.254,K,A*08
$GPGGA,080712.00,4759.26508,N,00750.25409,E,1,03,3.29,452.1,M,47.9,M,,*5D
$GPGSA,A,2,28,14,22,,,,,,,,,,3.44,3.29,1.00*06
$GPGSV,1,1,03,28,17,319,40,14,35,098,42,22,38,055,47*45
$GPGLL,4759.26508,N,00750.25409,E,080712.00,A,A*6B
$GPZDA,080712.00,09,04,2008,00,00*6D
$GPRMC,080713.00,A,4759.26512,N,00750.25401,E,0.110,25.73,090408,,,A*58
$GPVTG,25.73,T,,M,0.110,N,0.203,K,A*0F
$GPGGA,080713.00,4759.26512,N,00750.25401,E,1,03,3.29,452.1,M,47.9,M,,*5F
$GPGSA,A,2,28,14,22,,,,,,,,,,3.43,3.29,1.00*01
```



```
    ###Avarage GPS Coordinaten berechnen
    #Serialport
    avggpsserial = [gpsserial[1][1],0,0,0,0]

    for i in range(4):      # durchschnittsbreite berechnen
        avggpsserial[1] = avggpsserial[1] + gpsserial[i][1]
    avggpsserial[1] = avggpsserial[1] / 5

    for i in range(5):      # durchschnittslaenge berechnen
        avggpsserial[2] = avggpsserial[2] + gpsserial[i][2]
    avggpsserial[2] = avggpsserial[2] / 5

    for i in range(5):      # durchschnittsgeschwindigkeit berechnen
        avggpsserial[3] = avggpsserial[3] + gpsserial[i][3]
    avggpsserial[3] = avggpsserial[3] / 5

    for i in range(5):      # durchschnittsrichtung berechnen
        avggpsserial[4] = avggpsserial[4] + gpsserial[i][4]
    avggpsserial[4] = avggpsserial[4] / 5

##Avarage Values
    #Telnet
    avggpstelnet = [gpstelnet[0][0],0,0,0,0]
    for i in range(5):      # durchschnittsbreite berechnen
        avggpstelnet[2] = avggpstelnet[2] + gpstelnet[i][1]
    avggpstelnet[2] = avggpstelnet[2] / 5

    for i in range(5):      # durchschnittslaenge berechnen
        avggpstelnet[3] = avggpstelnet[3] + gpstelnet[i][2]
    avggpstelnet[3] = avggpstelnet[3] / 5

    for i in range(5):      # durchschnittstgeschwindigkeit berechnen
        avggpstelnet[4] = avggpstelnet[4] + gpstelnet[i][3]
    avggpstelnet[4] = avggpstelnet[4] / 5

    for i in range(5):      # durchschnittsrichtung berechnen
        avggpstelnet[5] = avggpstelnet[5] + gpstelnet[i][4]
    avggpstelnet[5] = avggpstelnet[5] / 5

##Umrechnung der Gradmase in Bogenmas
    avggpsserial[1]=avggpsserial[1]*math.pi/180
    avggpsserial[2]=avggpsserial[2]*math.pi/180
    avggpstelnet[1]=avggpstelnet[1]*math.pi/180
    avggpstelnet[2]=avggpstelnet[2]*math.pi/180

###Abstand Berechnen
    entfernung = math.acos(math.sin(avggpsserial[1])*math.sin(avggpstelnet[1])+
                 math.cos(avggpsserial[1])*math.cos(avggpstelnet[1])*
                 math.cos(avggpsserial[2]-avggpstelnet[2]))*6.371/1000

###Gebe Abstand und Richtung des Leaders in Datei schreiben

    result = open("result.txt", "w")
    result.write("Abstand: " + entfernung + "m" + "\n")
    result.write("Bewegungsrichtung des Leaders: " + avggpstelnet[4] + "m" + "\n")
```

Future possible Steps

- skripts make automatic availible at the startup of the gumstix
- upgrade of the skriptfile that only same dates will be compare
- change of the protokol from Telnet to FTP
    - parse and calculate the avarage directly on the root
    - run a FTP server on the gumstix to transmit the position of the leader
    - Retransmit the position on every clients that also clients out of range can receive that
    - clients make scan for the IP, on which they receive a position

    - advantages: - less network traffic because
      - Security can be im implement
      - Multiuser networks are possible

Referencelist:

- Our supervising tutor Chia Ching Ooi
- Prof. Schindelhauer - Mobile Ad Hoc Networks
- http://docwiki.gumstix.org/
- http://wikipedia.org
- http://www.remetter.de/os_linux_gumstix.php
- Python, das umfassende Handbuch, J. Ernesti, P. Kaiser
- Ram Ramanathan, Jason Redi. A Brief Overview of Ad Hoc Networks: Challenges and Directions
- Imrich Chlamtac, Marco Conti, Jennifer J.-N. Liu, Mobile Ad Hoc Networking: Imperatives and Challenges, in Ad Hoc Networks
- Wolfgang Kiess, Martin Mauve, A Survey on Real-World Implementions of Mobile Ad-Hoc Networks, in Ad Hoc Networks

Thank you

for your attention