

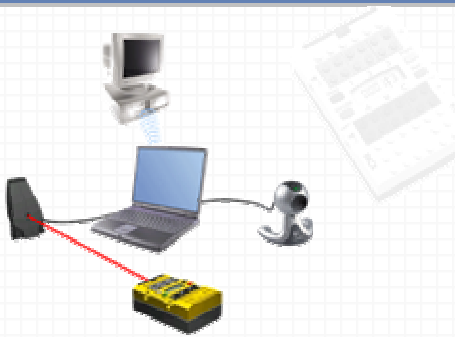
Robot-Kommunikation mit externen Systemen

Seminar Unterstützung von Landminendetektion durch Bildauswertungsverfahren und Robotereinsatz
 Patrick Arendt, Bent Jakuboski, Andreas Wulf – Universität Münster, WS 2003/2004

Die Referenten

Patrick Arendt
 Bent Jakuboski
 Andreas Wulf

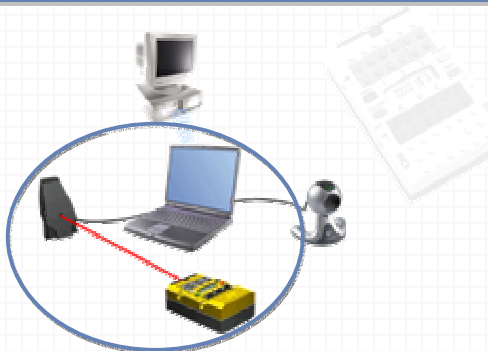
Der Kontext



1. Kommunikation zwischen RCX und Laptop (über IR-Tower)

Seminar Unterstützung von Landminendetektion durch Bildauswertungsverfahren und Robotereinsatz
 Patrick Arendt, Bent Jakuboski, Andreas Wulf – Universität Münster, WS 2003/2004

Einordnung in den Kontext



Lego Infrarot-Tower (1/2)

- Tower des Robotics Invention Systems 2.0 wird per USB mit dem Host verbunden
- Datenübertragung zwischen Tower und RCX erfolgt über gepulste Infrarotstrahlung
- Tower ist ein so genanntes Consumer Infrarot-Gerät (CIR Device)
- CIR wird normalerweise zur Fernbedienung von Endgeräten wie Fernseher, Videorekorder oder Hifi-Anlagen eingesetzt und ist bisher nicht standardisiert

Lego Infrarot-Tower (2/2)

- PCs, Laptops oder Handhelds verwenden hingegen das zur Datenübertragung standardisierte IrDA-Protokoll
- Frequenzen und Wellenlängen von CIR und IrDA sind nicht kompatibel zueinander
- Nachteile der Datenübertragung über Infrarot:
 - Sichtkontakt zwischen Geräten notwendig
 - Störanfällig durch andere Infrarotquellen oder (sichtbares) Licht
 - Geringe Übertragungsgeschwindigkeiten (beim IR-Tower standardmäßig 2400 Baud)

[7]

Programmierung des Towers (1/2)

- Zugriff auf den Lego Tower erfolgt unter Windows analog zum Öffnen einer Datei
- Mit Hilfe der API-Funktion "CreateFile" wird ein Handle erzeugt, mit dem sich der Tower ansprechen lässt:

```
TowerHandle = CreateFile(  
    "\\.\.\LEGOTOWER1",  
    GENERIC_READ | GENERIC_WRITE,  
    0, NULL,  
    OPEN_EXISTING,  
    0, 0);
```

- „CloseHandle“ gibt das Objekt-Handle wieder frei:

```
Success = CloseHandle(TowerHandle);
```

[8]

Programmierung des Towers (2/2)

- Mit "WriteFile" werden Bytes über den Tower versendet:

```
Success = WriteFile(  
    TowerHandle,  
    SendBuffer,  
    SendLength,  
    &BytesWritten, 0);
```

- Das Lesen einer Nachricht vom Tower erfolgt über "ReadFile":

```
Success = ReadFile(  
    TowerHandle,  
    RecBuffer,  
    1024,  
    &BytesRead, 0);
```

[9]

Kommunikationsprotokoll

Kommunikationsprotokoll

(abnehmendes Abstraktionsniveau)

- | |
|---------------------------|
| 1. Bytecode-Schicht |
| 2. Paketschicht |
| 3. Byte-Kodierungsschicht |
| 4. IR-Übertragungsschicht |

[10]

Kommunikationsprotokoll

Kommunikationsprotokoll

(abnehmendes Abstraktionsniveau)

- | |
|---------------------------|
| 1. Bytecode-Schicht |
| 2. Paketschicht |
| 3. Byte-Kodierungsschicht |
| 4. IR-Übertragungsschicht |

[11]

Bytecode

- Steuerung bzw. Programmierung des RCX erfolgt über (LASM) Bytecode
- Bytecode wird von Compilern (bspw. NQC) erzeugt oder kann auch von Hand generiert werden
- RCX-Firmware interpretiert Bytecode und reagiert entsprechend
- Bytecode-Instruktionen bestehen aus einem Opcode (1 Byte) und eventuell notwendigen Operanden

[12]

Bytecode-Beispiele (1/3)

- Test, ob RCX reagiert (Ping):

PC sendet:

0x10

Bytecode senden

Opcode

RCX antwortet:

0xEF

Antwort ist jeweils das Komplement der Anfrage

[13]

Bytecode-Beispiele (2/3)

- Sound Nr. 1 (Piepton) abspielen:

PC sendet:

0x51

0x01

Bytecode senden

Opcode

Operand

RCX spielt Sound und antwortet:

0xAE

Komplement des Opcodes

Mögliche Operanden:

0x00	Tastenklick
0x01	Piepton
0x02	fallender Ton
0x03	steigender Ton
0x04	Fehlerton
0x05	schnell steigender Ton

[14]

Bytecode-Beispiele (3/3)

- Batteriestand abfragen:

PC sendet:

0x30

Bytecode senden

Opcode

RCX antwortet:

0xCF

0xD8

0x1E

Komplement des Opcodes

B1

B2

- B1 ist das niederwertige Byte, B2 das höherwertige Byte des Ergebnisses
- Batteriestand = $B1 + B2 * 0xFF = 0x1EBA = 7866$ (mV)

[15]

Alternative Opcodes

- es ist nicht möglich, zweimal hintereinander die gleiche Nachricht zu senden
- **Grund:** RCX geht bei doppelter Sendung von einem Übertragungsfehler aus
- **Lösung:** zu (fast) jedem Opcode existiert ein alternativer Opcode
- Alternativer Opcode ergibt sich durch Addition von 0x08
- **Beispiel:** alternativer Opcode zu 0x10 lautet 0x18

[16]

Kommunikationsprotokoll

Kommunikationsprotokoll

(abnehmendes Abstraktionsniveau)

1. Bytecode-Schicht
2. Paketschicht
3. Byte-Kodierungsschicht
4. IR-Übertragungsschicht

[17]

Paketstruktur

- Bytecode-Instruktionen werden in Pakete verpackt
- Paketstruktur:

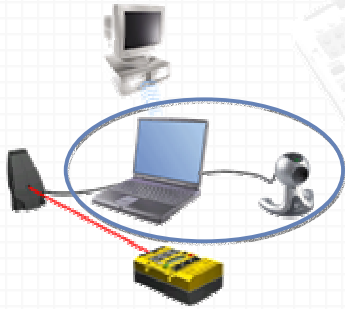
0x55 0xFF 0x00 D1 ~D1 D2 ~D2 ... Dn ~Dn C ~C

Paketkopf Datenbytes mit Komplementen Checksum mit Komplement

- Kopf des Pakets kündigt dem Empfänger das Folgen von Daten an
- Dx: Opcode (D1) oder Operand (D2, D3, ..., Dn)
- ~Dx: Komplement des Datenbytes
- C: Prüfsumme (Summe über aller Datenbytes ohne Übertrag)
- ~C: Komplement der Prüfsumme

[18]

Einordnung in den Kontext



[25]

Problemstellung

Die Webcam ist via USB mit dem Laptop verbunden

Wie kann man auf die Daten der WebCam zugreifen?

- Lösung: Multimedia-Frameworks
- Video for Windows
 - Microsoft DirectX
 - Java Media Framework

[26]

Gliederung

- | |
|-------------------------|
| 1. Video for Windows |
| 2. DirectX |
| 3. Java Media Framework |

[27]

Gliederung

- | |
|-------------------------|
| 1. Video for Windows |
| 2. DirectX |
| 3. Java Media Framework |

[28]

Was ist Video for Windows?

Schnittstellen zur *Aufnahme & Wiedergabe* von Audio- und Videoclips unter Windows

- 1992: Einführung des VFW-SDK
- Motivation: Windows-eigenes GDI (Windows Graphics Device Interface) für multimediale Anwendungen zu leistungsschwach
- später integriert in Windows 95 & NT 3.51
- Seit Windows 98 nicht mehr weiterentwickelt
- Wird in allen zukünftigen Windows-Versionen enthalten sein

[29]

AVICap-API

- Methoden zum Zugriff auf Videogeräte (bspw. eine Webcam) werden bei VFW in Form der AVICap-API bereitgestellt
- AVICap bietet eine einheitliche Schnittstelle, welche die Details der Kommunikation mit dem Videotreiber verbirgt
- Mit Hilfe von AVICap lassen sich sowohl Streaming-Videodaten als auch Einzelbilder aufzeichnen
- Auf den folgenden Folien werden die wichtigsten AVICap-Befehle vorgestellt

[30]

Aufnahmefenster

- Erstellen eines Aufnahmefensters:

```
CapHwnd = capCreateCaptureWindow(  
    "Capture-Fenster", // Fenstername  
    WS_CHILD | WS_VISIBLE, // Fensterstil  
    0, 0, // Fensterposition  
    320, 240, // Fensterdimensionen  
    ParentHwnd, // Handle des  
              // Elternfensters  
    0); // Child ID
```

- Falls erfolgreich, liefert die Funktion das Handle des erstellten Aufnahmefenster zurück

[31]

Verbindung mit Treiber

- Verbinden des Aufnahmefensters mit einem Videotreiber:

```
Success = SendMessage(CapHwnd,  
    WM_CAP_DRIVER_CONNECT, 0, 0L);
```

Index des Videotreibers (0-9)

- Eine Alternative zum Senden von Nachrichten ist die Verwendung von Makros - in diesem Fall:

```
Success = capDriverConnect(CapHwnd, 0);
```

- Aufgrund der besseren Lesbarkeit werden auf den folgenden Folien lediglich die Makros vorgestellt

[32]

Videodarstellung

- Mit Hilfe des capPreviewRate-Makros wird die Bildrate festgelegt
- Im folgenden Beispiel wird eine Anzeigedauer von 66ms pro Frame (ca. 15 Bilder/s) definiert:

```
capPreviewRate(CapHwnd, 66); // Rate in ms
```

- Mit dem capPreview-Makro lässt sich die Videodarstellung starten (und später mit Parameter FALSE auch wieder beenden):

```
capPreview(CapHwnd, TRUE); // Anzeige starten
```

[33]

Demonstration

- Bereits mit den vier bis hierhin vorgestellten Befehlen lässt sich eine Anwendung um die Darstellung eines Live-Videobildes ergänzen
- Praktische Demonstration mit Hilfe von VBA:



Webcam-Verbindung
herstellen

Webcam-Verbindung
trennen

[34]

Weitere Makros (1/2)

Weitere wichtige Makros zur Gewinnung von Einzelbildern:

- **capDriverGetCaps**: Abfrage der Fähigkeiten des Aufnahmetreibers (z.B. Overlay-Fähigkeit)
- **capGetStatus**: Ermittlung des Status des Aufnahmefensters (z.B. Dimensionen)
- **capGetVideoFormatSize**, **capGetVideoFormat**, **capSetVideoFormat**: Makros zur Ermittlung und Einstellung des Videoformats (z.B. Farbtiefe)
- **capDlgVideoSource**, **capDlgVideoFormat**, **capDlgVideoDisplay**: Anzeige von Dialogfenstern zur Konfiguration der Videoaufnahme

[35]

Weitere Makros (2/2)

- **capSetCallbackOnError**, **capSetCallbackOnStatus**, **capSetCallbackOnVideoStream**, **capSetCallbackOnFrame**: Makros zur Registrierung von Callback-Funktionen, die bei bestimmten Ereignissen aufgerufen werden
- **capGrabFrame**: Abfrage und Darstellung eines Einzelbildes
 - bei Aufruf werden die Bilddaten an die mit capSetCallbackOnFrame registrierte Callback-Funktion geschickt
- **capDriverDisconnect**: Trennen der Verbindung zwischen Aufnahmefenster und Videotreiber

[36]

Stärken & Schwächen

+

- schnell & einfach zu programmieren

-

- inkonsistentes Treibermodell
- Dateigröße auf 2 GB begrenzt
- beschränkter I/O-Durchsatz
- keine Synchronisierung
- keine Kontrolle über Videoqualitäts-Merkmale wie Helligkeit, Kontrast, etc

[37]

... aus unserer Sicht

+

- schnell & einfach zu programmieren

-

- keine Kontrolle über Videoqualitätsmerkmale wie Helligkeit, Kontrast, etc.

[38]

Gliederung

1. Video for Windows
2. DirectX
3. Java Media Framework

[39]

Was ist DirectX?

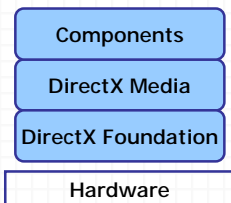
Sammlung von APIs zur Programmierung von Multimediaanwendungen unter MS Windows, die sich durch

- ✓ schnellen Hardwarezugriff &
- ✓ Geräteunabhängigkeit auszeichnen

- 1995: Einführung von DirectX
- Motivation:
 - Bestehende Multimedia-Schnittstellen weisen konzeptionelle Schwächen auf
- Basiert auf dem Component Object Model (COM)

[40]

Schichtenarchitektur



Drei-Schichten-Architektur:

Components

- Setzt auf Funktionen der Media- & Foundation-Schicht auf

DirectX Media

- High-Level APIs
- Benutzt Foundation-Schicht

DirectX Foundation

- Low-level APIs
- Verbirgt Hardware

[41]

Hardwareunabhängigkeit

Abstrahieren von verwendeter Hardware durch Zwei-Schichten-Prinzip:

- **Hardware Abstraction Layer (HAL)**
 - spricht vorhandene Hardware direkt an
- **Hardware Emulation Layer (HEL)**
 - emuliert nicht vorhandene Hardware

[42]

DirectShow

Für uns interessant:

DirectShow

[43]

Architektur (1/2)

• Filtergraph:

verkettete Anordnung austauschbarer Module (Filter)

- Source Filter: liest Daten
- Transform Filter: transformiert Daten
- Rendering Filter: gibt Daten aus

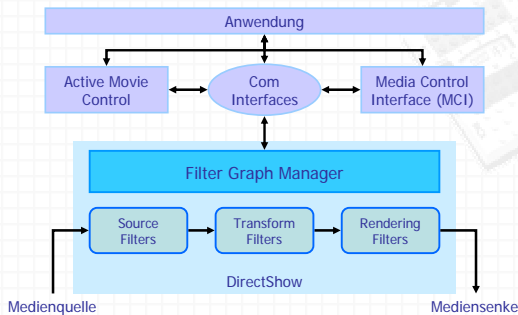
• Filter Graph Manager

koordiniert den Datenfluss und lässt sich über Schnittstellen direkt ansprechen:

- COM Interface
- Active Movie Control (Media Player Control)
- Media Control Interfaces (MCI)

[44]

Architektur (2/2)

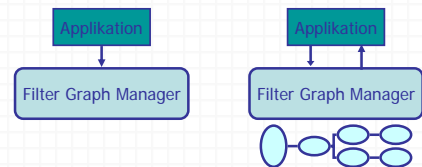


[45]

Programmierung

Schritte bei der Programmierung einer DirectShow-Anwendung:

- Erzeugen einer Instanz des Filter Graph Managers
- Auswahl und Anordnung der jeweiligen Filter zu einem Filter Graph mittels des FGM
- Datenstreaming durch den Filter Graph und Kontrolle



[46]

Gliederung

1. Video for Windows
2. DirectX
3. Java Media Framework

[47]

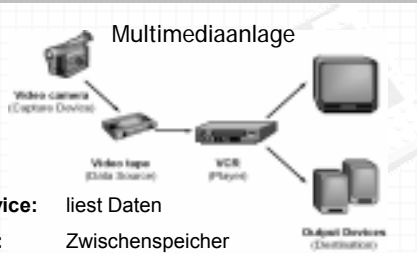
JMF - Was ist das?

API für die Integration von zeitabhängigen Medien in Java-Applets und -Applikationen

- Capture
 - Processing
 - Presentation
- 1997: JMF 1.0 von Intel, Sun & Silicon Graphics
 - 1998: Rückzug von Intel
 - 1999: JMF 2.0 von Sun & IBM
 - 2000: JMF 2.1

[48]

High-Level-Architektur



- Capture Device:** liest Daten
- DataSource:** Zwischenspeicher
- Player:** Daten abspielen, aufnehmen, bearbeiten
- Output Devices:** Ausgabegeräte

[49]

Programmier-Vorgehen

- Ansprechen der Webcam via CaptureDeviceManager
- Erzeugen eines CaptureDeviceInfo-Objekts
- Übergabe dessen an einen Player / Processor
 - Player -> Wiedergabe
 - Processor -> Wiedergabe & Weiterverarbeitung

[50]

VfW vs. DirectX vs. JMF

	VfW	DirectX	JMF
Proprietät	closed-source	closed-source	open-source
Akzeptanz <small>(für prof. Anwendungen)</small>	hoch	Sehr hoch	≈ Null
Plattform	Windows	Windows	plattform-unabhängig
Hardware-abhängigkeit	bedingt unabhängig	unabhängig	bedingt unabhängig
Performance	hoch	Sehr hoch	Java-Niveau
Programmiersprachen	VB, VC++, C, Delphi u.v.a	VB, VC++, C, Delphi u.v.a	Java
Komplex	gering	sehr hoch	hoch

[51]

Die API unserer Wahl

Welche Schnittstelle eignet sich für unsere Zwecke am Besten?

unter Windows:

1. Video for Windows

bei hohen Anforderungen an Bildweiterverarbeitung:

2. DirectX

unter Linux oder Java als bevorzugte Programmiersprache:

3. Java Media Framework

[52]

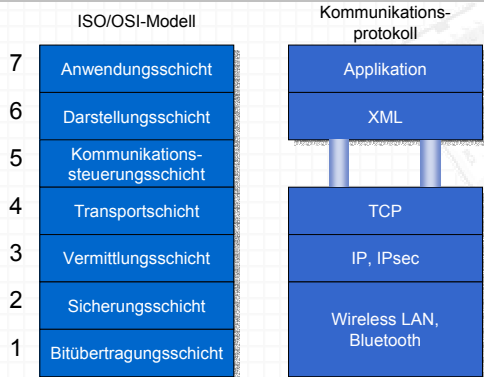
3. Kommunikation zwischen Laptop und Host-Rechner

Einordnung in den Kontext



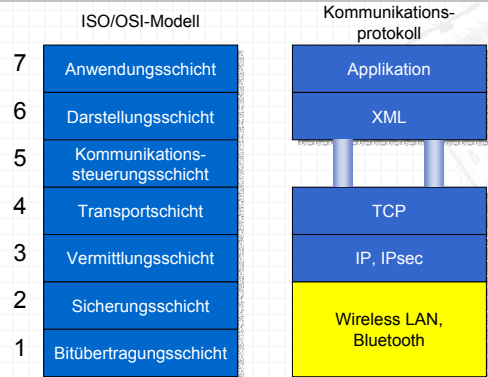
[54]

Gliederung



[55]

Gliederung



[56]

Gliederung

1. Geschichte
2. Bluetooth
3. Wireless LAN

[57]

Gliederung

1. Geschichte
2. Bluetooth
3. Wireless LAN

[58]

Geschichte

- ca. 150 b.c.: Rauchsignale
- 1884: Beweis des Wellencharakters der elektrischen Übertragung durch den Raum
- 1915: erste drahtlose Sprachübertragung
- 1997: Wireless LAN – IEEE802.11
- 1999: IEEE802.11b & Bluetooth

[59]

Gliederung

1. Geschichte
2. Bluetooth
3. Wireless LAN

[60]

Definition & Entstehung

Standard zur Datenübertragung per Funk im ISM (Industrial, Scientific, Medical) - Band

1994 Studie von Ericsson mit dem Ziel einer Funktechnologie mit folgenden Eigenschaften:

- Herstellerunabhängig
- Billige Schnittstelle
- Geringer Energieverbrauch
- Integrierte Sicherheitsfunktionen

1998 Gründung der Bluetooth Special Interest Group (SIG)

2000 Bluetooth Standard 1.0

[61]

„Bluetooth“?



- Vorbild: dänischer König Harald Blåtand („Blauzahn“)
- Vereinte 983 a.d. Königreiche Dänemark und Norwegen

- Bluetooth vereinheitlicht die Kommunikation zwischen Geräten unterschiedlicher Hersteller

[62]

Bitübertragung

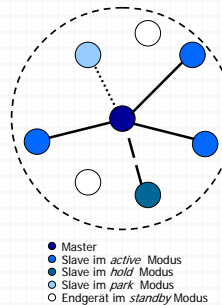
Frequenzbereich: 2,402 – 2,48 GHz,
unterteilt in 79 Kanäle mit 1 MHz Abstand

Zwei Sendeklassen für Endgeräte:

- 10 Meter Reichweite bei 2,5 mW Sendeleistung
- 100 m Reichweite bei 100 mW Sendeleistung

[63]

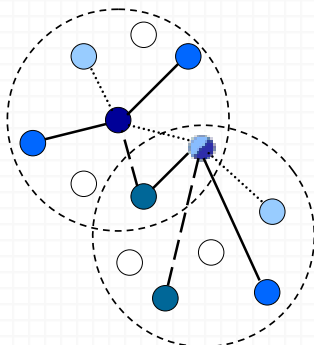
Topologie - Piconet



- Ein Master und bis zu sieben Slaves bilden zusammen ein ‚Piconet‘
- Piconets entstehen automatisch & dynamisch
- Master bestimmt Verbindungsart & -geschwindigkeit, verwaltet die Adressen und sendet Synchronisationssignale

[64]

Topologie - Scatternet



- Ein Scatternet besteht aus bis zu 10 sich überlappenden Piconets
- Ein Slave des einen Piconets kann im anderen Master sein

[65]

Vor- & Nachteile von Bluetooth

+

- geringe Hardwarekosten
- relativ störungsresistent
- gut geeignet für Echtzeitanwendungen wie Sprachübertragung
- geringer Stromverbrauch
- relativ sicher

-

- noch in der Einführungsphase
- relativ langsam
- ohne verstärkte Sendeleistung nur geringe Reichweite

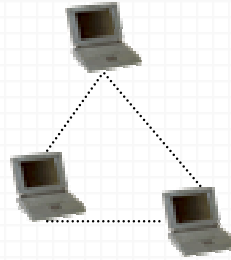
[66]

Gliederung

1. Geschichte
2. Bluetooth
3. Wireless LAN

[67]

Topologie – Ad-hoc Netzwerk



Vorteile

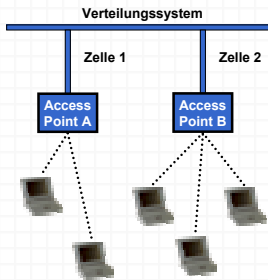
- Einfach & günstig
- Mobil
- Routing möglich

Nachteile

- Reichweite begrenzt
 - Geringe Bandbreite
 - Erreichbarkeit nicht-transitiv
- Geeignet für spontane oder mobile Anwendungen (z.B. Robotereinsatz)

[68]

Topologie – Infrastruktur-Netzwerk



Vorteile

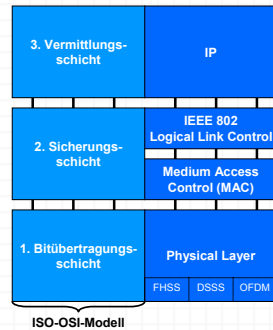
- Große Reichweite
- Hohe Bandbreite
- Definierte Erreichbarkeit

Nachteile

- Festinstallation erforderlich
 - Hoher Administrationsaufwand
- Geeignet für Gebäudevernetzung

[69]

Protokollarchitektur



- WLAN definiert die ersten beiden Schichten des ISO-OSI-Modells
- IP setzt direkt darauf auf

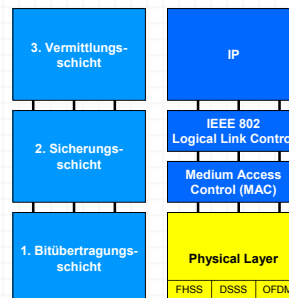
[70]

Das MA-Problem

- Das Multiple Access Problem besteht, wenn mehrere Benutzer das gleiche Frequenzband benutzen.
- Auf der *Physical Layer*: verschiedenartige Benutzer, z. B.
 - Verschiedene WLAN-Standards
 - Mikrowellenöfen
- Auf der *MAC Layer*: verschiedene Benutzer
 - Unterschiedliche Stationen eines WLANs

[71]

Bitübertragungsschicht



- Zur Lösung des MA-Problems:
Betrachtung der Physical Layer

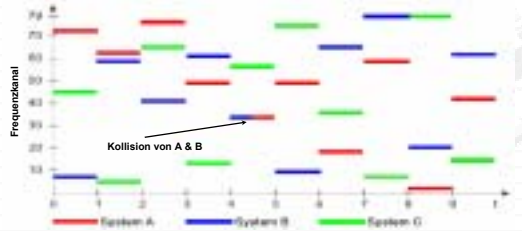
[72]

Übertragungstechniken

- Frequency Hopping Spread Spectrum (FHSS)
- Direct Sequence Spread Spectrum (DSSS)
- Orthogonal Frequency Division Multiplexing (OFDM)

[73]

FHSS



- Mindestens 2,5 Frequenzwechsel pro Sekunde
- 79 Kanäle mit 1MHz Bandbreite für max. 13 FHSS-Systeme

[74]

DSSS

Signalspreizung durch Pseudonoise (PN)-Code:

Chipping

- **Sender:** XOR der Daten mit dem Chip-Code
- Jedes Bit wird so zu einer Chip Sequenz
Übertragung mit bis zu 11 Mbps
- **Empfänger:** Daten XOR Code XOR Code =
originäre Daten
- Bei teilweiser Zerstörung der Chip-Sequenz ist
unter Umständen das originale Bit erkennbar

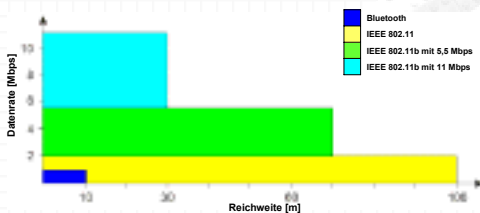
[75]

OFDM

- Frequency Division Multiplexing (FDM):
ein high-rate Datenstrom wird in mehrere lower-
rate Ströme geteilt, welche parallel über
mehrere Frequenzen gesendet und dann
multiplex werden
- Sind die Einzelfrequenzen orthogonal, d. h.
unabhängig, spricht man von OFDM
- Überlappungen sind dann zugelassen, hierbei
liegt jeweils eine Nullstelle im Maximum der
anderen Einzelfrequenz
- Einsparung von 50% im Vergleich zu FDM

[76]

Datenrate vs. Reichweite

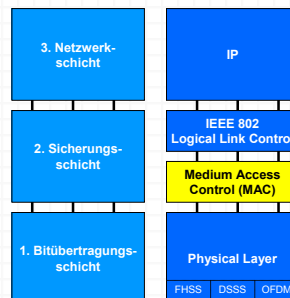


Je höher die Datenrate,

- desto geringer die Reichweite bei gleicher Leistung
- desto höher die Störanfälligkeit

[77]

Sicherungsschicht



Zur Lösung des MA-
Problems:

Jetzt Betrachtung der
Medium Access Control

[78]

Medium Access Control

- Kollisionserkennung
- Kollisionsvermeidung
- Reservierung

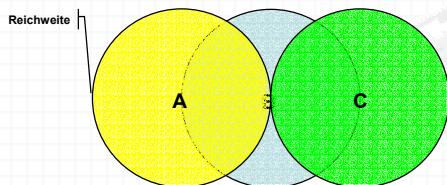
[79]

Mögliche Probleme

- „Verstecktes“ Endgerät
- „Ausgeliefertes“ Endgerät
- „Near/Far Problem“

[80]

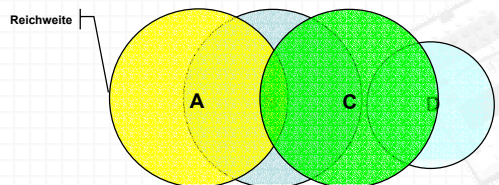
„Verstecktes“ Endgerät



- A sendet zu B, C empfängt A nicht
- C will zu B senden, Medium ist für C frei
- Kollision bei B
- A ist „versteckt“ für C

[81]

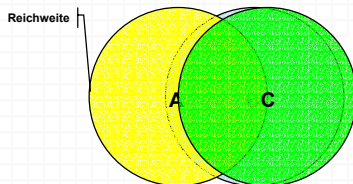
„Ausgeliefertes“ Gerät



- B sendet zu A, C will zu D senden
- C muss warten, da ein „besetztes“ Medium signalisiert wird
- Da A außerhalb der Reichweite von C ist, ist dies unnötig
- C ist B „ausgeliefert“

[82]

„Near/Far Problem“



- A und B senden an C
- Da die Signalstärke quadratisch mit der Entfernung abnimmt, „übertönt“ das Signal von B das andere
- C kann A nicht hören
- Exakte Leistungskontrolle notwendig

[83]

Problemlösungen

Verschiedene Verfahren:

- Code Division Multiple Access (CDMA)
- Frequency Division Multiple Access (FDMA)
- Space Division Multiple Access (SDMA)
- Time Division Multiple Access (TDMA)
 - bei WLAN benutzt

[84]

TDMA

- Sendezeiten werden in Schlitze aufgeteilt, die anforderungsgesteuert oder fest vergeben werden
- Teilnehmer sind nacheinander aktiv
- Etabliert
- Voll digital
- Vielfältig einsetzbar
- Synchronisation und Schutzzeiten erforderlich

[85]

TDMA - Verfahren

Verfahren zur Vergabe von Zeitschlitzen:

- Aloha
- Slotted Aloha
- Explizite Reservierung
- Implizite Reservierung
- Reservation TDMA

Zusätzlich wird oft „Multiple Access with Collision Avoidance“ verwendet, d. h. es werden kurze Pakete mit Senderadresse, Empfängeradresse und Paketgröße gesendet.

[86]

Bisher veröffentlichte Standards

	a	b	g
Bitrate brutto:	54 MBit/s	11 MBit/s	54 MBit/s
Bitrate netto:	32 MBit/s	5 MBit/s	32 MBit/s
Reichweite:	15 - 50 m max. bis 2 km	30 - 100 m max. bis 3 km	30 - 100 m max. bis 3 km
Frequenzbereich:	5,4 GHz	2,4 - 2,4835 GHz	2,4 - 2,4835 GHz
Sendeleistung:	30 mW	100 mW	100 mW
Empfohlenes Einsatzgebiet:	Indoor (kollidiert mit Radaranlagen)	Indoor / Outdoor	Indoor / Outdoor
Besonderheit:	inkompatibel zu b & g	am Weitesten verbreitet	abwärtskompatibel zu b

[87]

Die Qual der Wahl

Bluetooth oder WLAN?

Welche Technologie ist für uns erste Wahl?

[88]

Bluetooth - einfacher & billiger?

- Bluetooth nie als Netzwerkersatz gedacht, sondern zur drahtlose Anbindung von Peripherie über Kurzstrecken
 - minimaler Stromverbrauch
 - einfache Verbindungen
- Konsequenzen in der Praxis:
 - Reichweite viel anfälliger für unebenes Gelände als WLAN (Freie Sichtverbindung reicht nicht: „Fresnel-Zone“)
 - Übertragungsrate fällt mit zunehmender Reichweite viel stärker ab als bei WLAN
 - 100 Meter Reichweite für unsere Zwecke utopisch

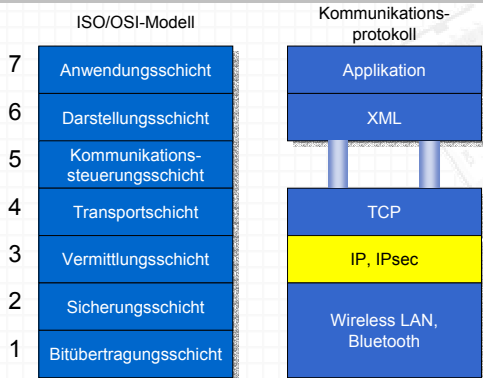
[89]

WLAN - Technologie unserer Wahl

- Für unsere Zwecke ist WLAN klar im Vorteil:
 - + Reichweite
 - + Bandbreite
 - + Störanfälligkeit
- Gleichzeitiger Einsatz von Bluetooth nicht zu empfehlen:
 - gleiches Frequenzband wie 802.11b & 802.11g
 - gegenseitige Störungen sind wahrscheinlich

[90]

Gliederung



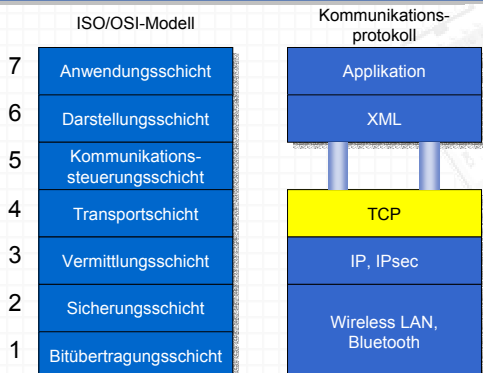
[91]

IP, IPsec

- IP:
 - Das Internet Protocol (IP) kümmert sich um den **Transport von Datenpaketen** (sogenannte Datagramme) vom Sender zum Empfänger
 - Hauptaufgaben von IP: **Adressierung** von Rechnern sowie das **Fragmentieren** bzw. **Reassemblieren** der Datenpakete
- IPsec:
 - IP bietet keine Sicherheit
 - IPsec ergänzt IP um **Authentifizierung**, **Integritätskontrolle** und **Verschlüsselung**

[92]

Gliederung



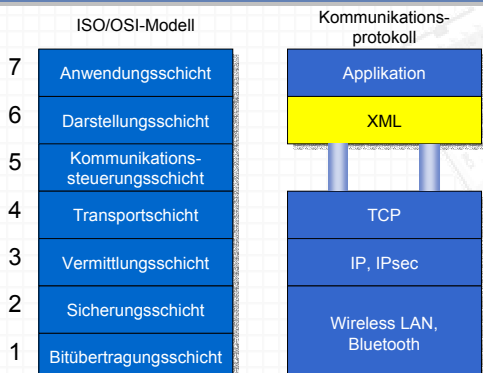
[93]

TCP

- Aufgabe des Transmission Control Protocols (TCP) ist die **Bereitstellung eines sicheren und zuverlässigen Transports** von Daten durch ein Netzwerk
- TCP bietet Fehlererkennung und -korrektur auf dem gesamten Übertragungsweg

[94]

Gliederung



[95]

Darstellungsschicht

- Aufgabe der Darstellungsschicht ist die Kodierung und Darstellung der auszutauschenden Informationen
- Wahl von XML (Extensible Markup Language) zur Datenformatierung
- Beispiel eines in XML "verpackten" Datenpakets:

```
<?xml version="1.0"?>
<!DOCTYPE rcml SYSTEM "rcml.dtd">
<rcml>
  <header>
    <sender>Host1</sender>
    <receiver>Client1</receiver>
    <uuid>98e262ac-24cf-12d6-c13d-010032b29093</uuid>
  </header>
  <cmd>
    <runmotor motor="Motor_A" duration="100" />
  </cmd>
</rcml>
```

[96]

Document Type Definition (DTD)

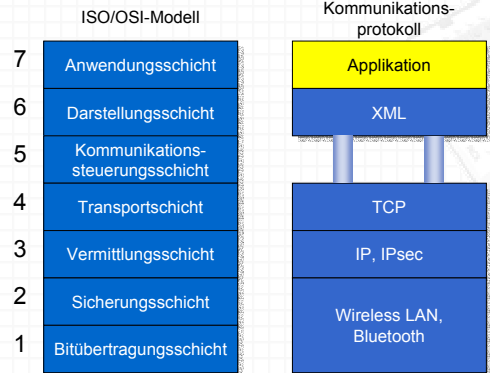
- Verschachtelung und Benennung der XML-Tags wird von DTD (Document Type Definition) vorgegeben:

```
<ELEMENT rcm1 (header, (cmd|reply))>
<ELEMENT header (sender, receiver, replyto?, uuid)>
<ELEMENT sender (#PCDATA)>
<ELEMENT receiver (#PCDATA)>
<ELEMENT replyto (#PCDATA)>
<ELEMENT uuid (#PCDATA)>
<ELEMENT cmd (runmotor|stopmotor|setmotordirection|setmotorpower|getsensordata|
getbatterylevel|getwebcamimage|sendbytecode|ping|startprogram|transferprogram|
powerdown)>
<ELEMENT runmotor EMPTY>
<!ATTLIST runmotor motor (Motor_A|Motor_B|Motor_C) #REQUIRED duration CDATA #REQUIRED>
...
<ELEMENT reply (error|sensordata|bytecodereply|batterylevel|pingreply|
webcamimage)>
<ELEMENT error (#PCDATA)>
<ELEMENT sensordata (#PCDATA)>
<!ATTLIST sensordata sensor (Sensor_0|Sensor_1|Sensor_2) #REQUIRED>
...
<ELEMENT imgdata (#PCDATA)>
<!ATTLIST imgdata format (BMP|JPG) #REQUIRED>
<ELEMENT objdata (distance, center, radius)>
<ELEMENT distance (#PCDATA)>
<ELEMENT center (#PCDATA)>
<ELEMENT radius (#PCDATA)>
```

- Mit Hilfe der DTD kann die übertragende XML-Struktur validiert werden

[97]

Gliederung



[98]

Anwendungsschicht

- Die Anwendungsschicht stellt dem Anwendungsprozess eine Schnittstelle zu den Diensten der Darstellungsschicht zur Verfügung
- Die Anwendung hat ausschließlich über die Schicht 7 Zugang zur Protokollhierarchie
- Im konkreten Fall ist die Anwendungsschicht eine API, die auf den XML-Parser der Darstellungsschicht zugreift

[99]

4. Praktische Anwendung

Verba docent, exempla trahunt.

Seminar Unterstützung von Landminendektection durch Bildauswertungsverfahren und Robotereinsatz
Patrick Arendt, Bent Jakuboski, Andreas Wulf – Universität Münster, WS 2003/2004

Hardware: Lego-Roboter



[101]

Probleme bei der Konstruktion

- Ein RCX mit zwei oder drei Motoren ist zu schwach, um das Gewicht des Laptops zu tragen
- **Lösung:** zwei RCX mit insgesamt vier Motoren (verteilt auf vier Ketten)
- Beide RCX werden über einen IR-Tower angesprochen
- **Problem dabei:** IR-Antworten der RCX überlagern sich und machen eine Auswertung der Antworten unmöglich

[102]

Fotos (1/3)



[103]

Fotos (2/3)



[104]

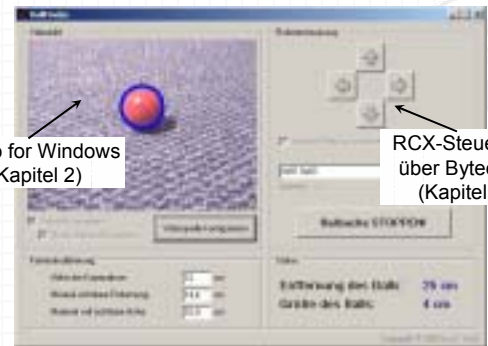
Fotos (3/3)

Roboter auf
Nahrungssuche



[105]

Software: BallFinder



Video for Windows
(Kapitel 2)

RCX-Steuerung
über Bytecode
(Kapitel 1)

[106]

Fragen?



[107]

Danke für die Aufmerksamkeit!



[108]