CGRA-Versuch

Aufgabe:

Optimierung der Akquisition eins GPS Empfängers für einen auf dem AMIDAR-Modell basierenden Java-Prozessor

- Zwei Lösungsaspekte
  - Maximale Performance: Je weniger Takte, desto besser
  - Minimaler Energieverbrauch: Je weniger Energieverbrauch, desto besser
- Nochmal: Die Qualität der Lösung kann die Note der Prüfung beeinflussen!

Organisatorisches

- Abgabe: 26.2.2018

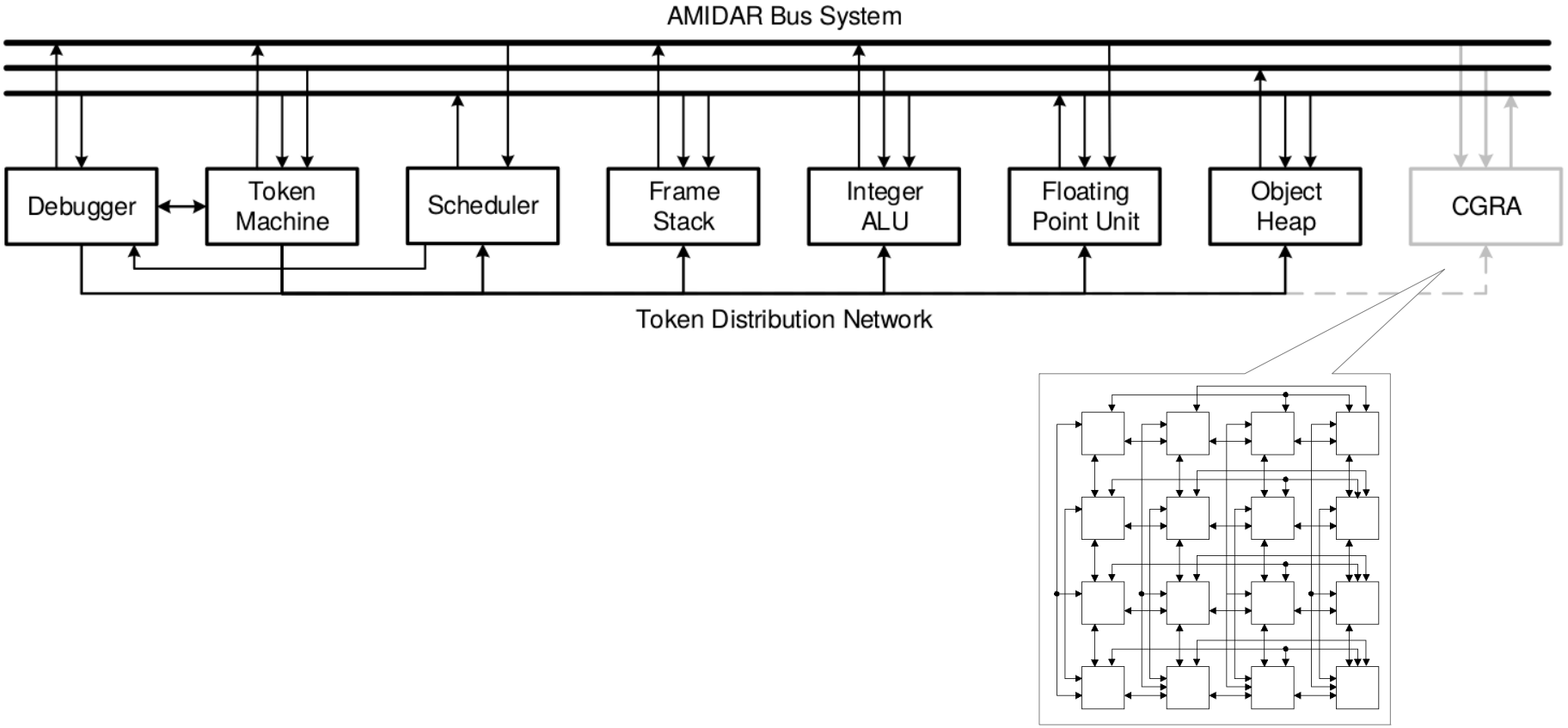
- Betreuer:
  - Grundsätzlich erst mal die Tutoren, aber wenn alles nichts hilf:
  - Johanna Rohde (rohde@rs.tu-darmstadt.de)
    - Aufgabenstellung
  - Lukas Jung (jung@rs.tu-darmstadt.de)
    - AMIDAR

AMIDAR (1)

- **A**daptive **M**icro-**I**nstruction **D**riven **AR**chitecture
- Grundlegende Idee
  - Abbildung rechenintensiver Operationen auf CGRA
  - Betrachtung nur von Schleifen (Loops)
- Vorteil:
  - Synthetisieren neuer Funktionseinheiten zur Laufzeit
    - ➔ **dynamische Adaption**
  - Selbe Hardware für unterschiedliche Anwendungen

AMIDAR (2)

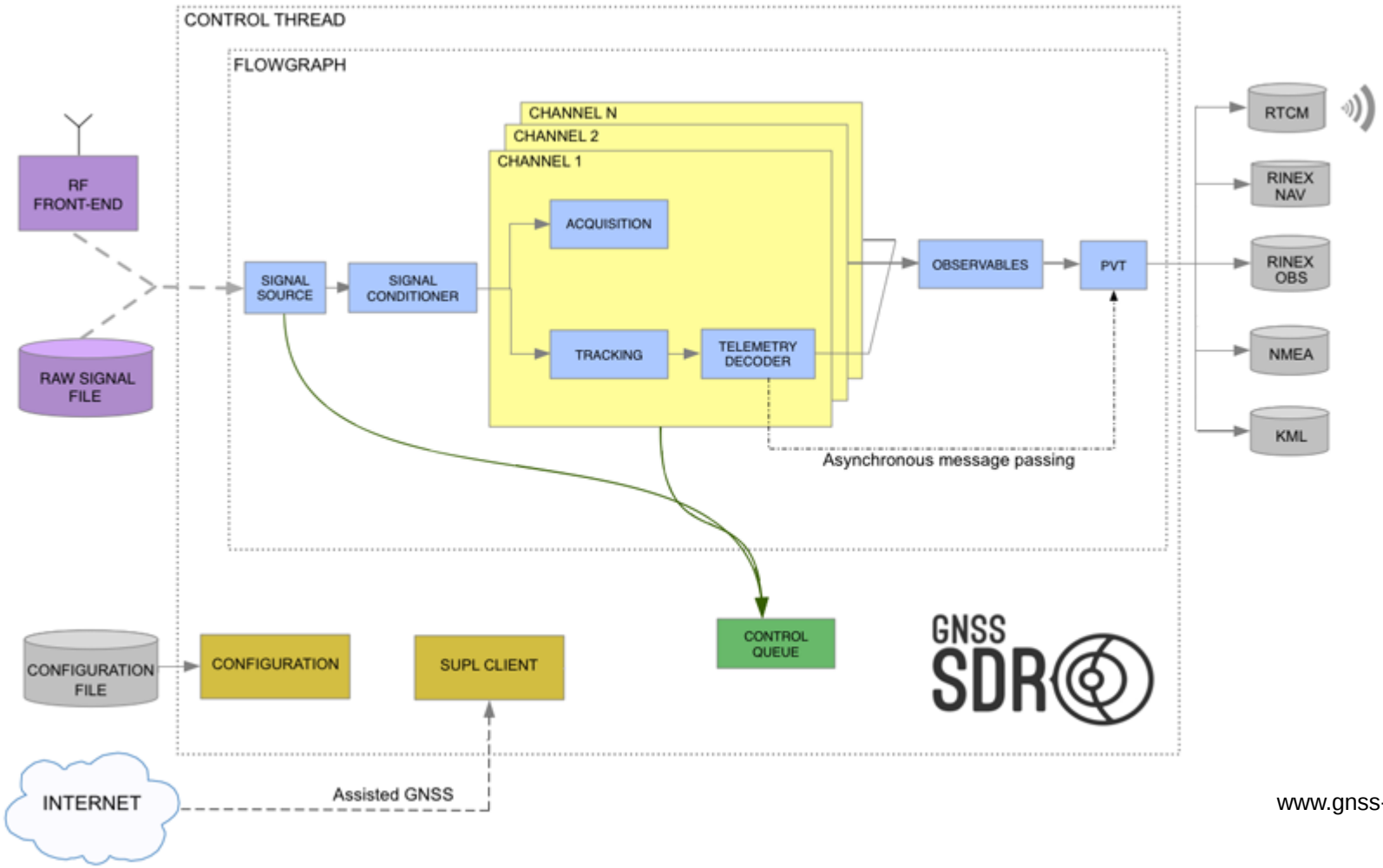
- Aufbau eines AMIDAR-Prozessors



Software Defined Radio

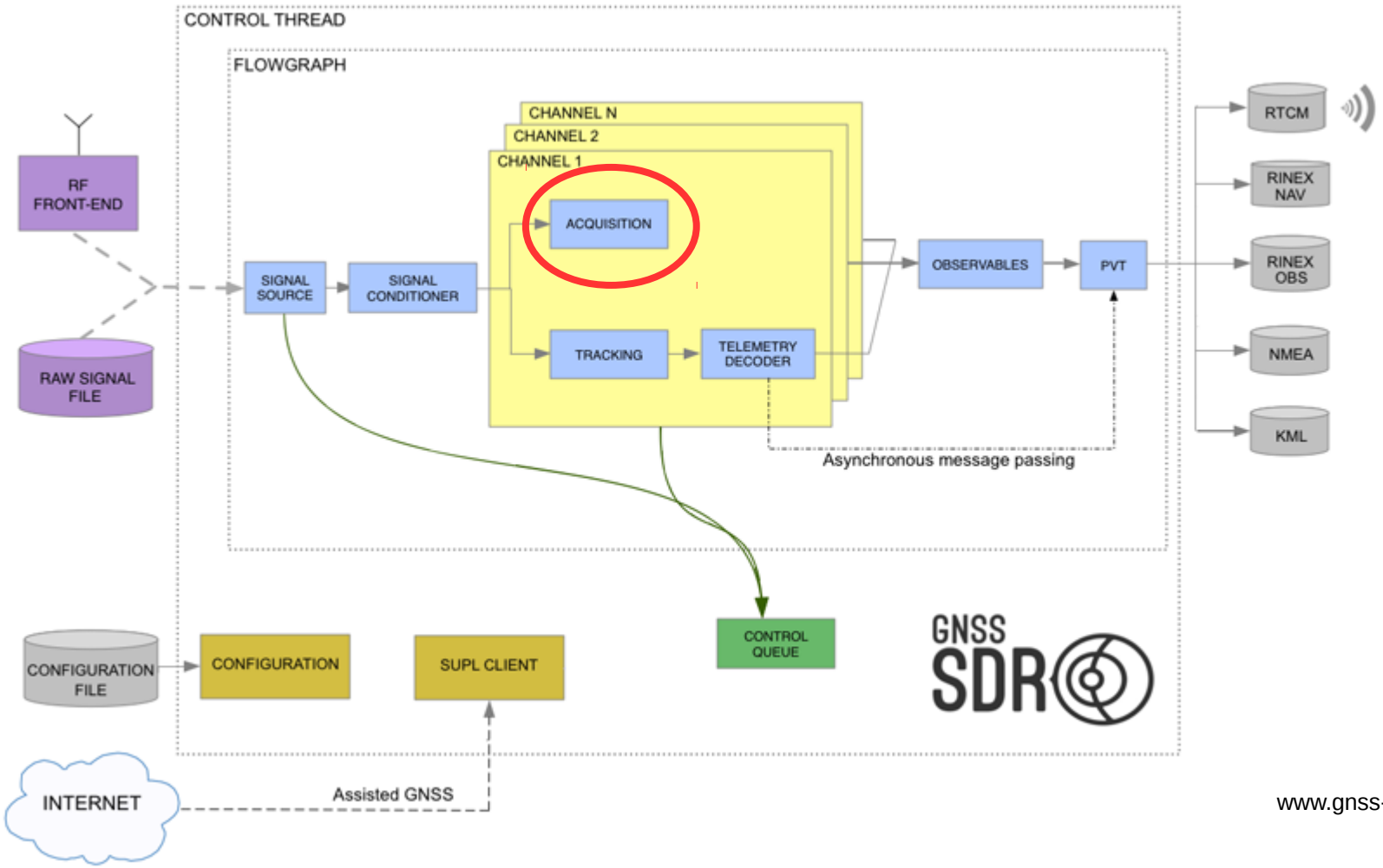
- Hochfrequenz-Sender/-Empfänger aus Software
- Analoges Signal wird direkt digitalisiert und in Software verarbeitet
- Beispiel: GPS-Empfänger
  - L1 Frequenz (1575,42 MHz)
  - C/A Code (für jeden Satelliten unterschiedlich)

GPS-SDR



www.gnss-sdr.org

GPS-SDR

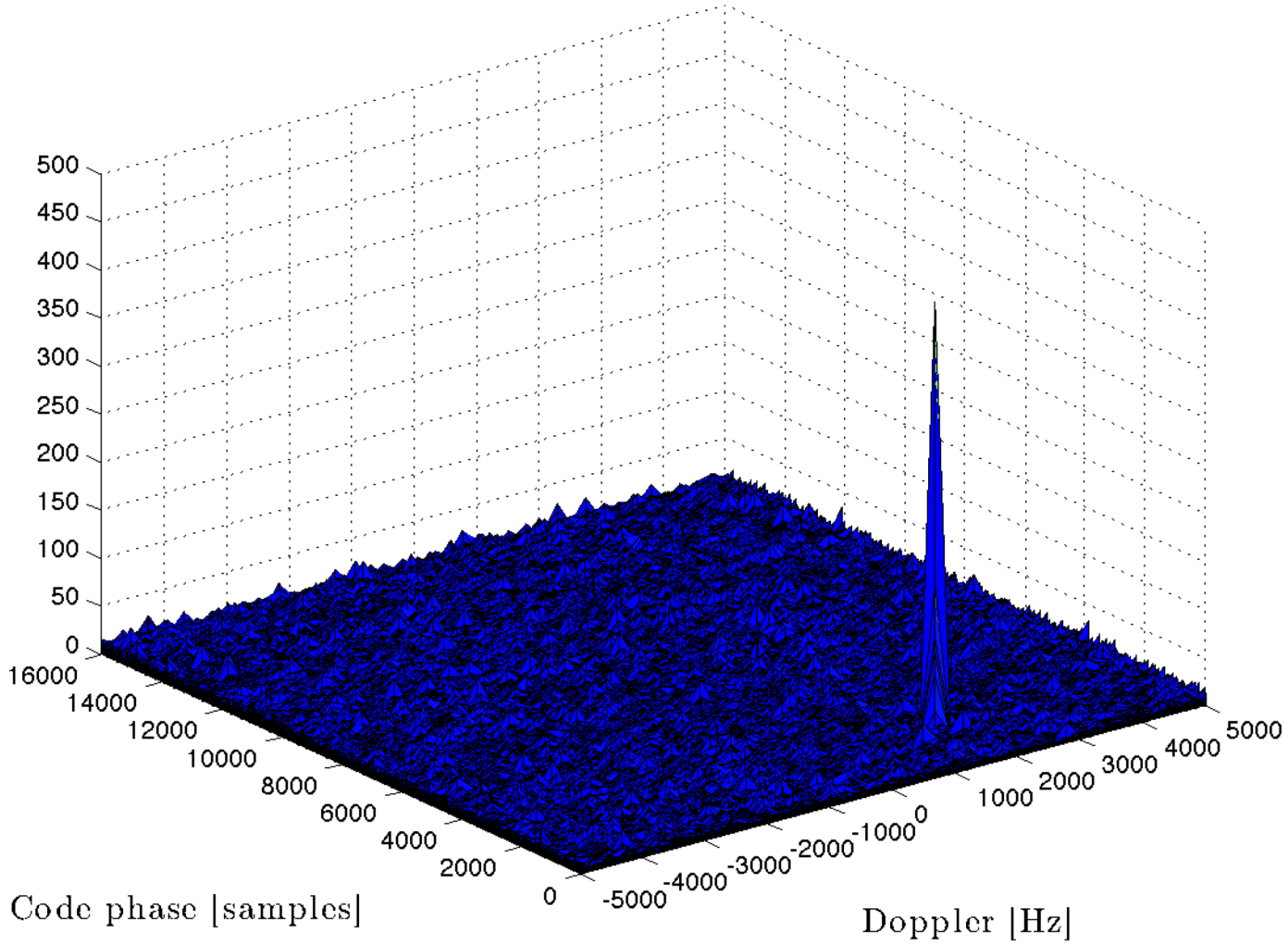


www.gnss-sdr.org

Akquisition (1)

- Berechnet für jeden Satelliten:
  - Wurde das (L1 C/A) Signal eines Satelliten empfangen?
  - Dopplerverschiebung
    - … für welche Dopplerverschiebung wird die Signalstärke am größten
  - Codeverschiebung
    - … wo fängt mein Signal an
- Bekommt das Signal als komplexwertige I/Q Samples übergeben
- Dient als Initialisierung für das Tracking

Akquisition (2)



http://www.cttc.es/wp-content/uploads/2013/03/121208-2582419-fernandez-9099698438457074772.pdf

Inputs

- f_s = Abtastfrequenz
- f_{step} = Frequenzabstand
- f_{max} = Maximale Frequenz
- f_{min} = Minimale Frequenz
- $m = \lfloor (f_{max} - f_{min}) / f_{step} \rfloor + 1$ = Anzahl aller Frequenzen
- $F \in \mathbb{Z}^m = \{ (f_{min} + i \cdot f_{step}) | 0 \leq i < m \}$ = Menge aller Frequenzen
- $\gamma \in \mathbb{R}$ = Grenzwert für die Akquisition

Konfiguration

- $N \in \mathbb{Z}$ = Anzahl der Samples = Samples pro Code
- $C \in \mathbb{C}^N$ = GPS L1 C/A Codes
- $X_{\text{in}} \in \mathbb{C}^N$ = Eingangssample

Eingangswerte

Outputs

- $f_{doppler}$ = Dopplerverschiebung
- τ = Codeverschiebung
- Akquisitionsnachricht $\in$ (true, false)

Algorithmus

$$X \in \mathbb{C}^{mxN} = \{X_f \in \mathbb{C}^N | f \in F\}$$

$$X_{f_d}[n] = X_{\text{in}}[n] \cdot e^{-(j2\pi f_d n / f_s)} \quad \text{for n = 0, .. , N-1; } f_d \in F$$

$$R \in \mathbb{C}^{mxN} = \{r_{f_d} \in \mathbb{C}^N | f_d \in F\}$$

← Komplexe Ergebnismatrix

$$r_{f_d} = \frac{1}{N} IDFT\left[DFT(X_{f_d}) \cdot \left[[DFT(C)]^*\right]^T\right] \quad \text{for } f_d \in F$$

$$S_{max} \in R$$

$$\{S_{max}, f_{doppler}, \tau\} = \max_{f,\tau} |R(f,\tau)|^2$$

← Maximalwert

$$P_{\text{in}} = \frac{1}{N} \sum_{n=0}^{N-1} |X_{\text{in}}[n]|^2$$

← Geschätzte Signalleistung

$$\Gamma = \frac{S_{\max}}{P_{\text{in}}}$$

```
if ( Γ > γ)
    Akquisitionsnachricht = true;
else
    Akquisitionsnachricht = false;
```

Algorithmus

$$X \in \mathbb{C}^{mxN} = \{X_f \in \mathbb{C}^N | f \in F\}$$

$$X_{f_d}[n] = X_{\text{in}}[n] \cdot e^{-(j2\pi f_d n / f_s)} \quad \text{for n = 0, .. , N-1; } f_d \in F$$

$$R \in \mathbb{C}^{mxN} = \{r_{f_d} \in \mathbb{C}^N | f_d \in F\}$$

$$r_{f_d} = \frac{1}{N} IDFT\left[DFT(X_{f_d}) \cdot \left[[DFT(C)]^*\right]^T\right] \quad \text{for } f_d \in F$$

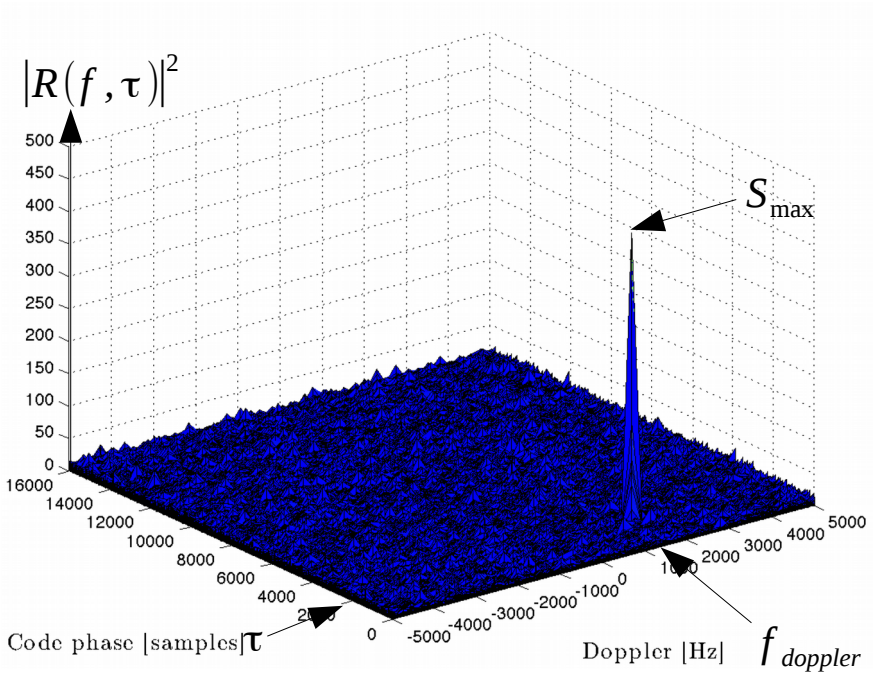
$$S_{max} \in R$$

$$\{S_{max}, f_{doppler}, \tau\} = \max_{f,\tau} |R(f,\tau)|^2$$

$$P_{\text{in}} = \frac{1}{N} \sum_{n=0}^{N-1} |X_{\text{in}}[n]|^2$$

$$\Gamma = \frac{S_{\max}}{P_{\text{in}}}$$

if (Γ > γ)
    Akquisitionsnachricht = true;
else
    Akquisitionsnachricht = false;



Implementierung (1)

- Schreiben Sie eine Klasse „*GpsAcquisition*“
- Interface:
  - *Acquistion(int nrOfSamples)*
    - Konstruktor, der die Klasse mit Anzahl der Samples
  - *void enterSample(float real, float imag);*
    - Nimmt jeweils ein komplexes Sample entgegen
    - Speichert dieses intern
  - *void enterCode(float real, float imag);*
    - Nimmt jeweils einen komplexen Wert des L1 C/A Codes entgegen
  - *boolean startAcquisition();*
    - Führt Akquisition aus
    - Gibt die Akquisitionsnachricht zurück
    - Speichert Doppler- und Codeverschiebung intern
  - int getDopplerverschiebung(); int getCodeverschiebung()
    - Auslesen der Doppler- und Codeverschiebung

Implementierung (2)

- Hinweise
  - Die Präzision *float* ist ausreichend
  - Verwenden sie einen Konstruktor, der N – die Anzahl der Samples - übergeben bekommt
  - Sampledatensätze können in Moodle runter geladen werden
  - Zum Optimieren könnt ihr N reduzieren und f_{step} erhöhen um die Laufzeit zu reduzieren
  - Tipp: Schaut euch nochmal das Rechnen mit komplexen Zahlen an!

Konfiguration

- f_s = 400 kHz
- f_{step} = 1 kHz
- f_{max} = 5 kHz
- f_{min} = -5 kHz
- Für N gilt: $N \in \mathbb{Z}$ und N ist teilbar durch 16
- γ = 0.015

Hausaufgabe

- Installieren Sie die benötigte Software um den AMIDAR Simulator nutzen zu können
  - ➔ http://www.rs.tu-darmstadt.de/fileadmin/AmidarRS2.zip
- Implementieren Sie die Akquisition in Java
  - Java 1.4
- Die Optimierung folgt in einem zweiten Schritt nächste Woche (Optimierungstechniken werden nächste Woche vorgestellt)