

Travaux d'initiatives personnelles encadrés

# **Contrôle des surfaces par ultrasons**

Année 2017-2018

# Plan

## I - Réalisation d'un système de contrôle

Description du système

Détermination de ses caractéristiques

## II - Contrôle de surface

Contrôle par pulses et temps de réponse

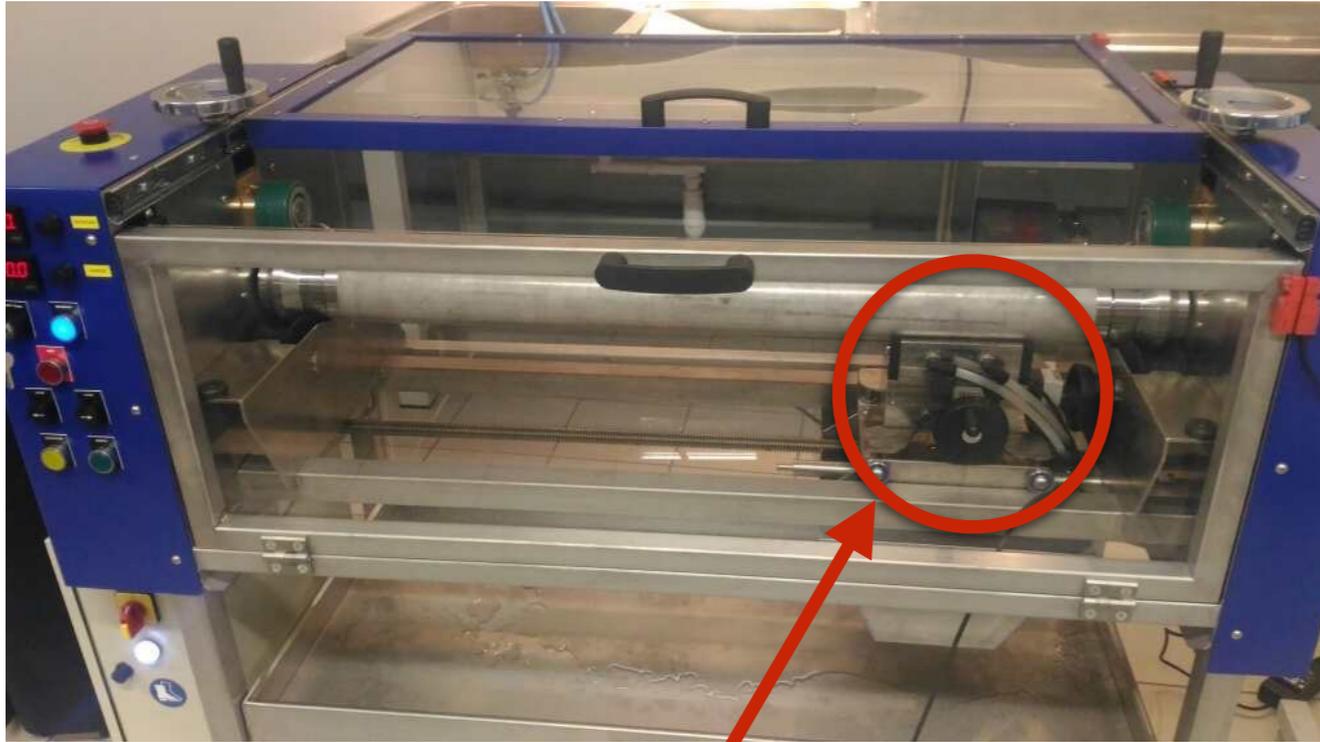
Contrôle grâce aux ondes stationnaires

Reconstruction de la surface

## Conclusions

## Annexes

## Systeme industriel

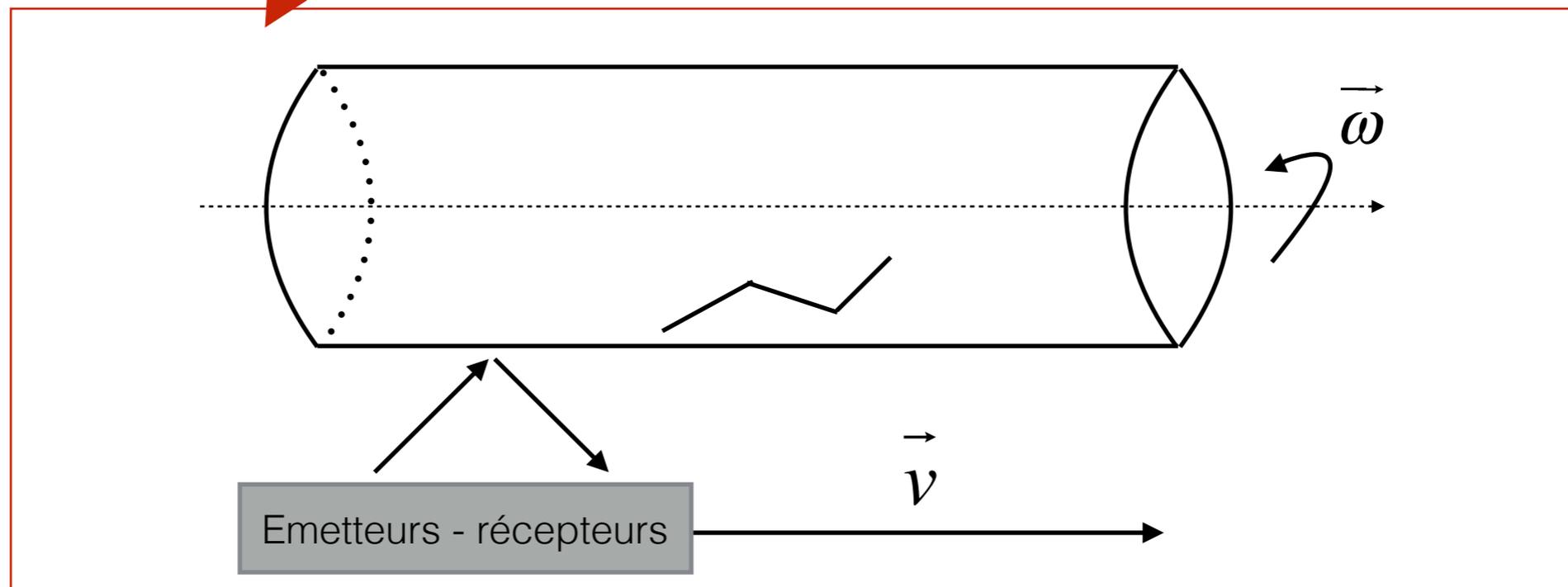


Systeme industriel (GRETA de Montbard)

Constructeur : ENRH

Fréquence de travail : 10 MHz

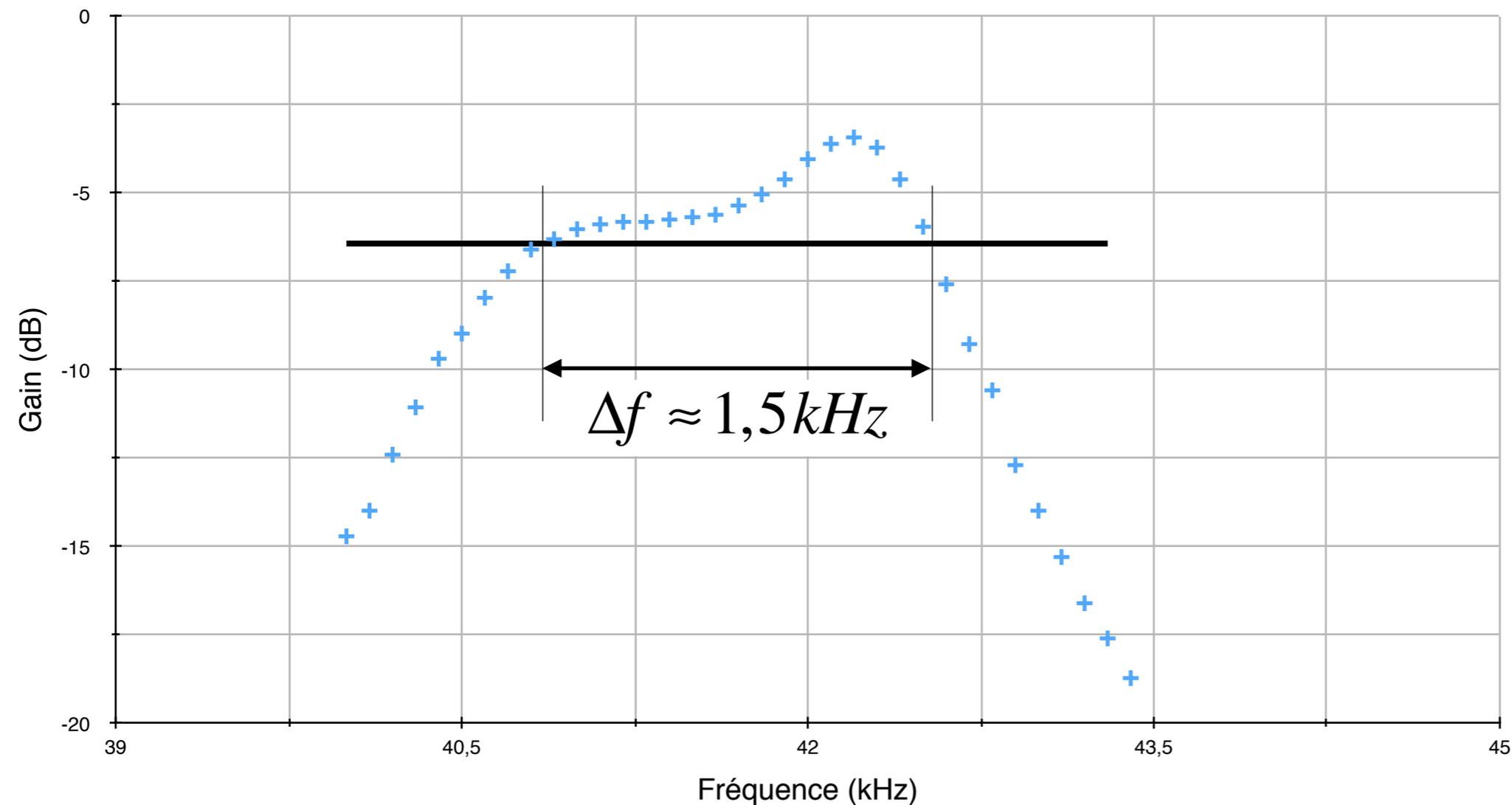
Précision : 5/10<sup>ème</sup> de millimètre



# Caractéristique de notre système

## Diagramme de Bode

Diagramme de Bode du système émetteur - récepteur



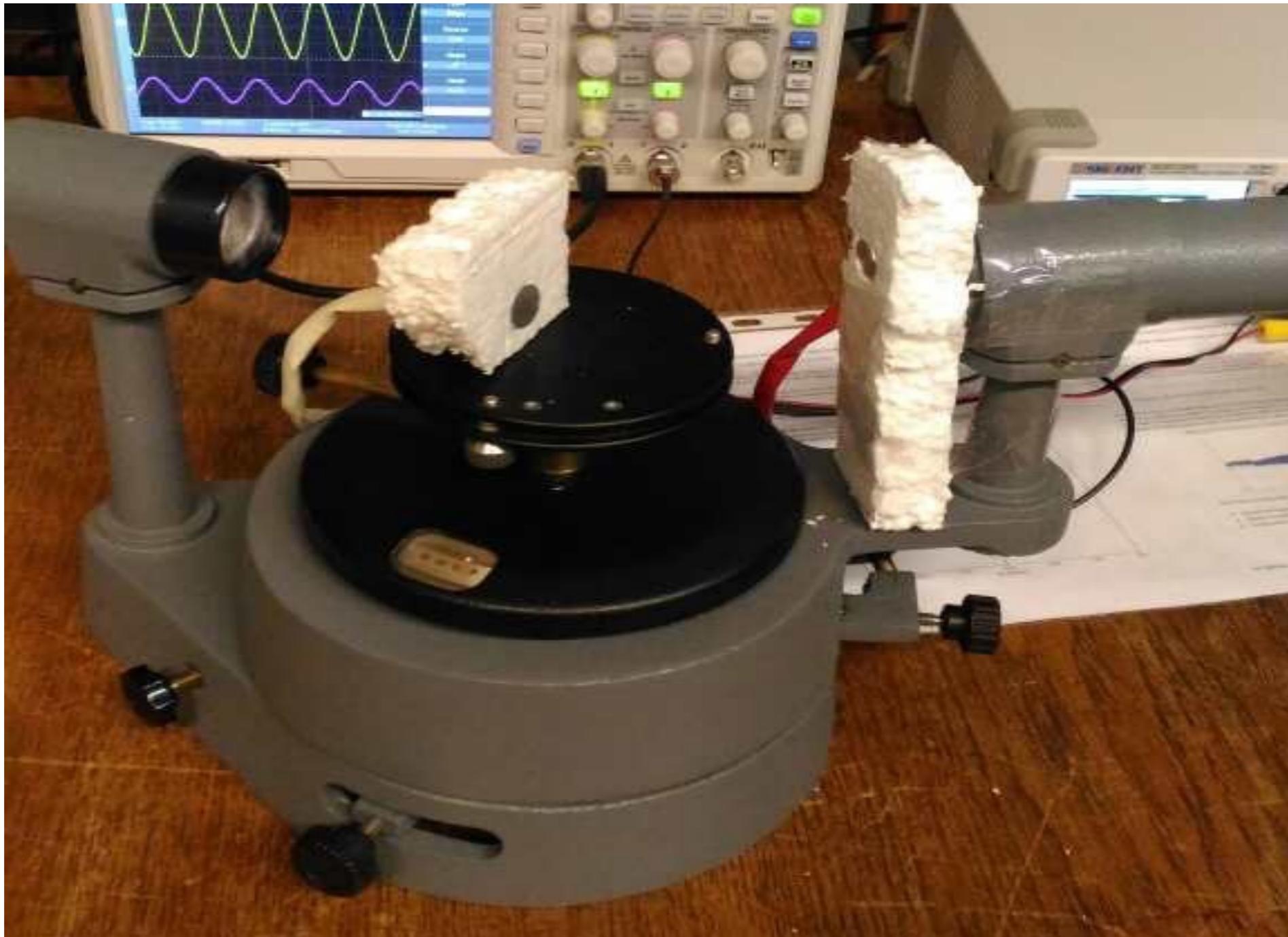
Zhipu

$$f_{\text{résonance}} = (42,2 \pm 0,1) \text{ kHz}$$

$$\Rightarrow Q = \frac{f_{\text{résonance}}}{\Delta f} \approx 30$$

# Caractéristique de notre système

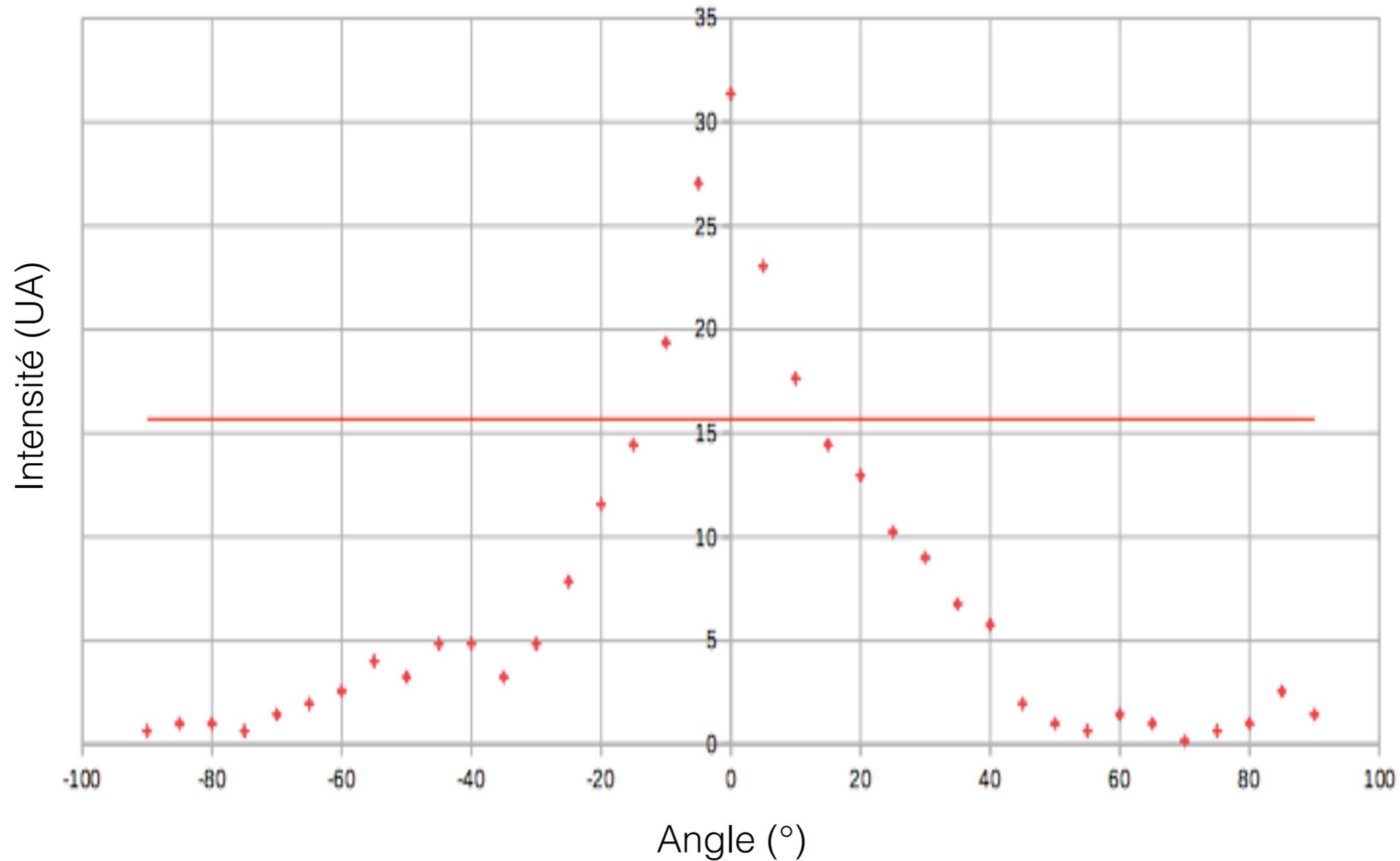
## Diagramme de rayonnement



# Caractéristique de notre système

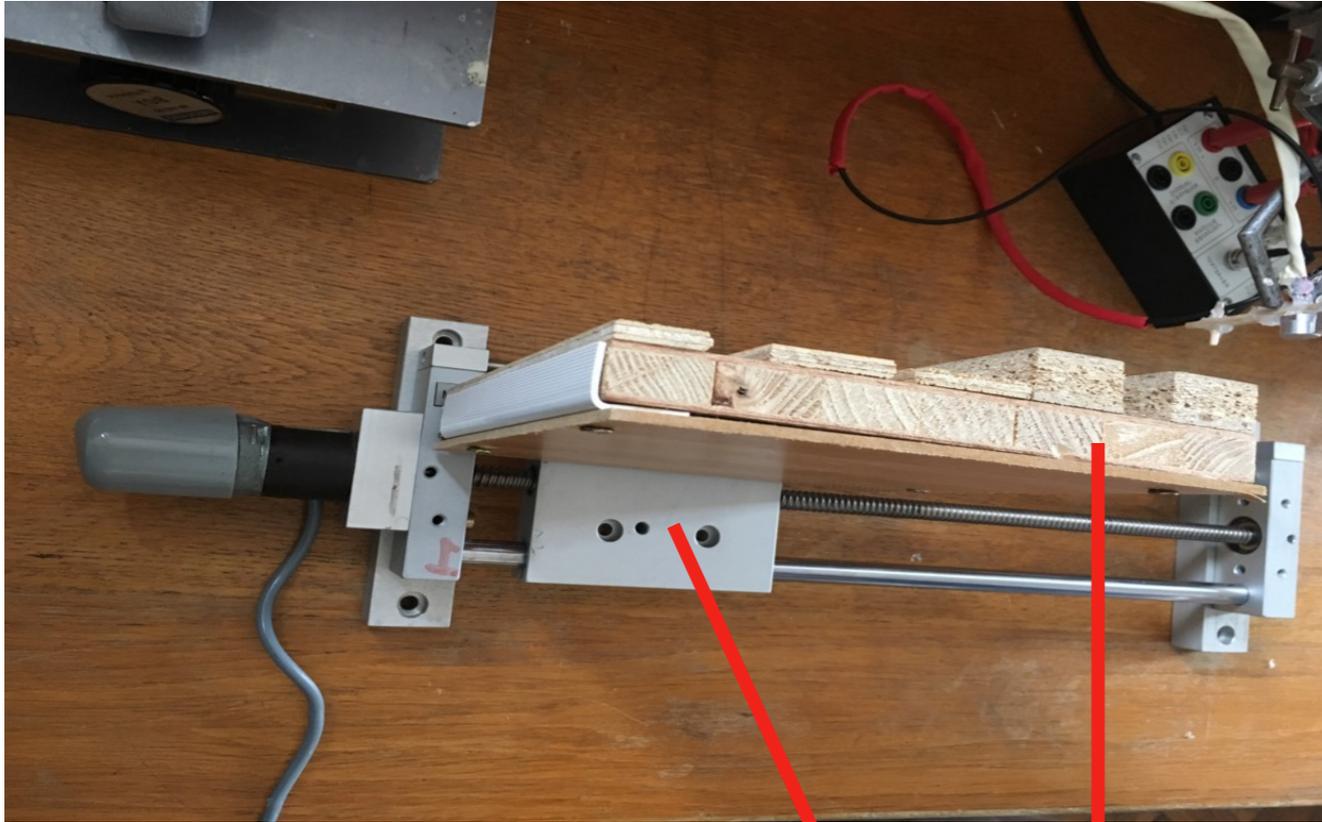
## Diagramme de rayonnement

Intensité en fonction de l'angle



$$\frac{I_{\max}}{2} : 30^{\circ}$$

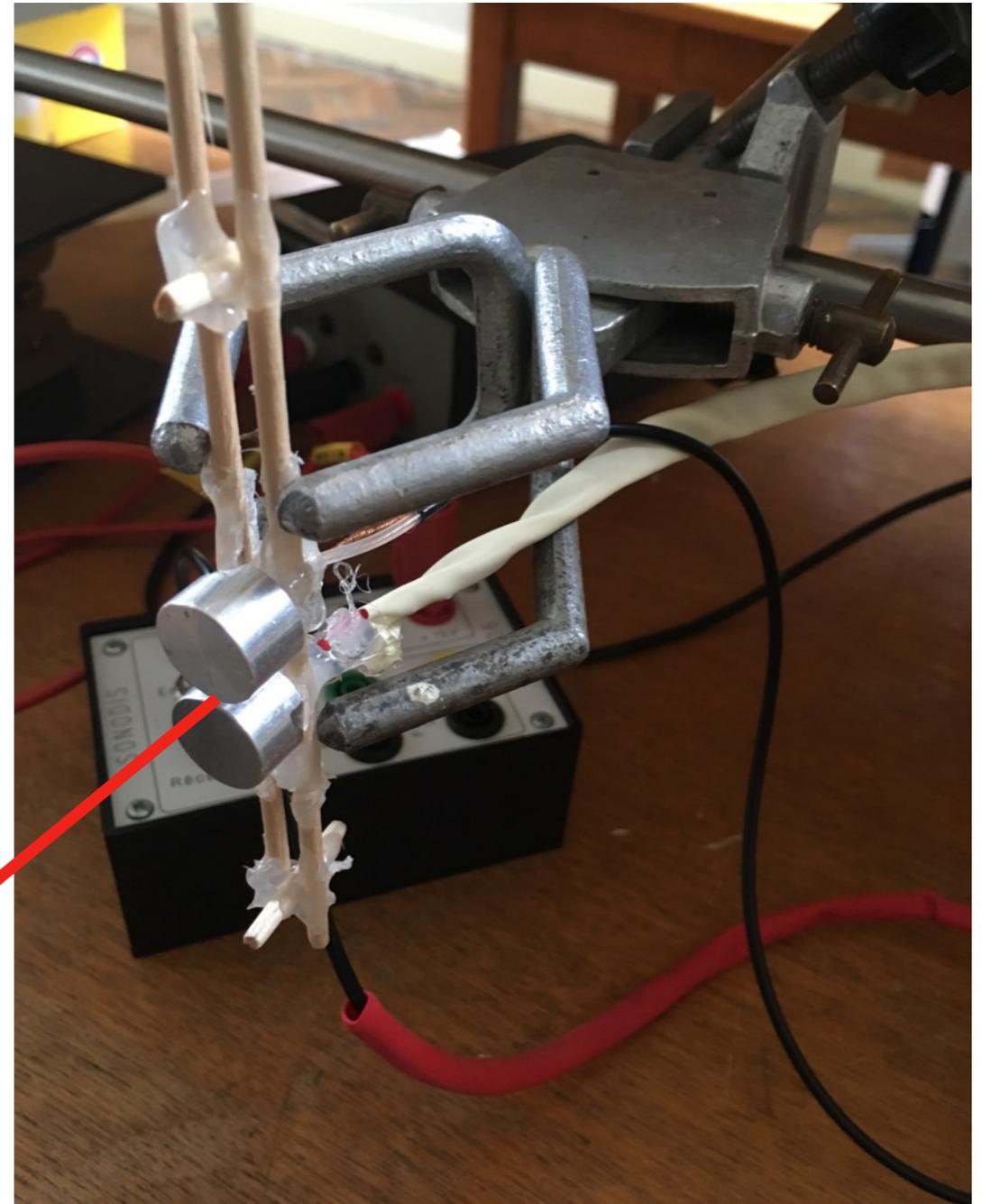
## Description du système



Plateau se translatant

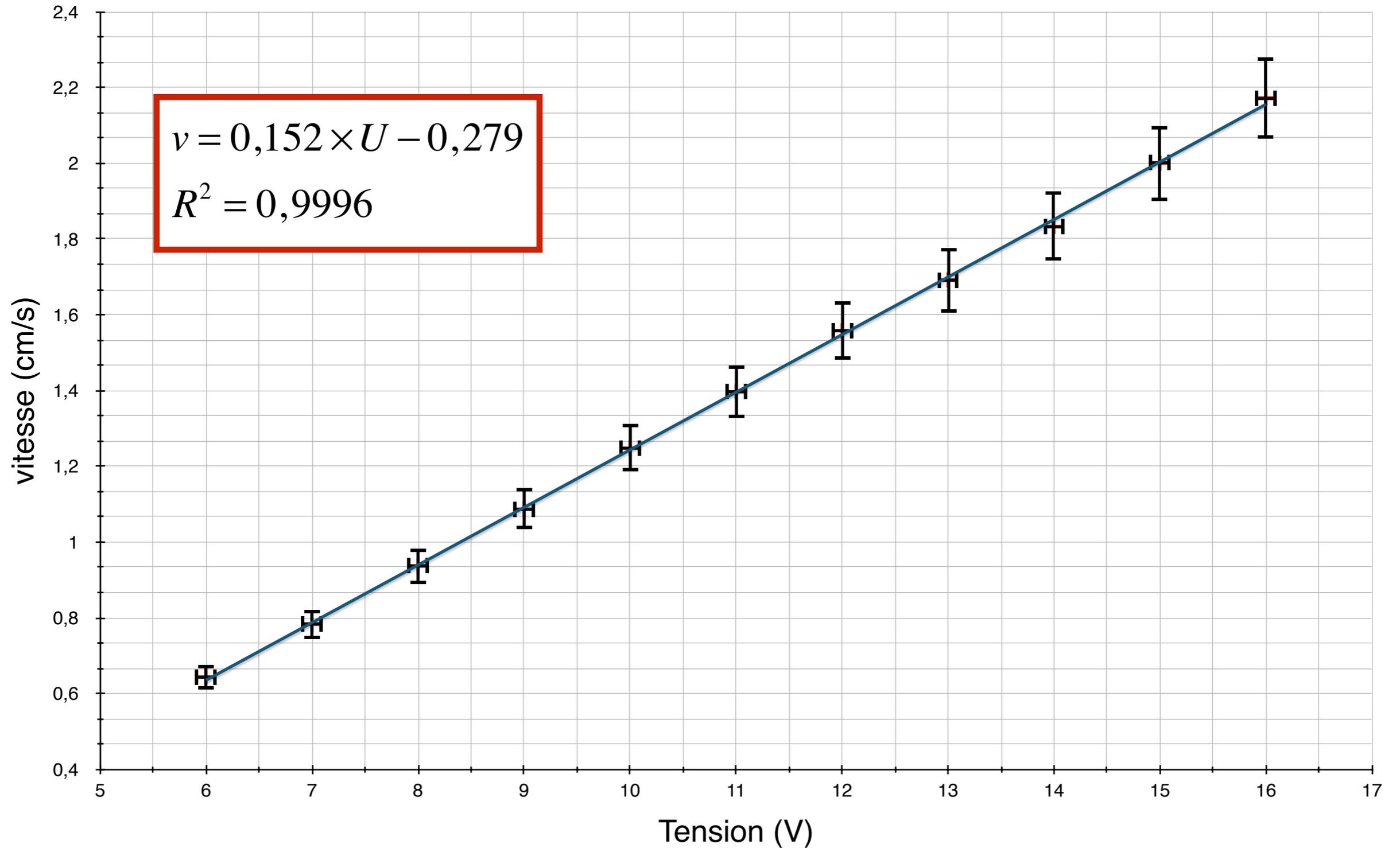
Planche de test

Système émetteur-récepteur



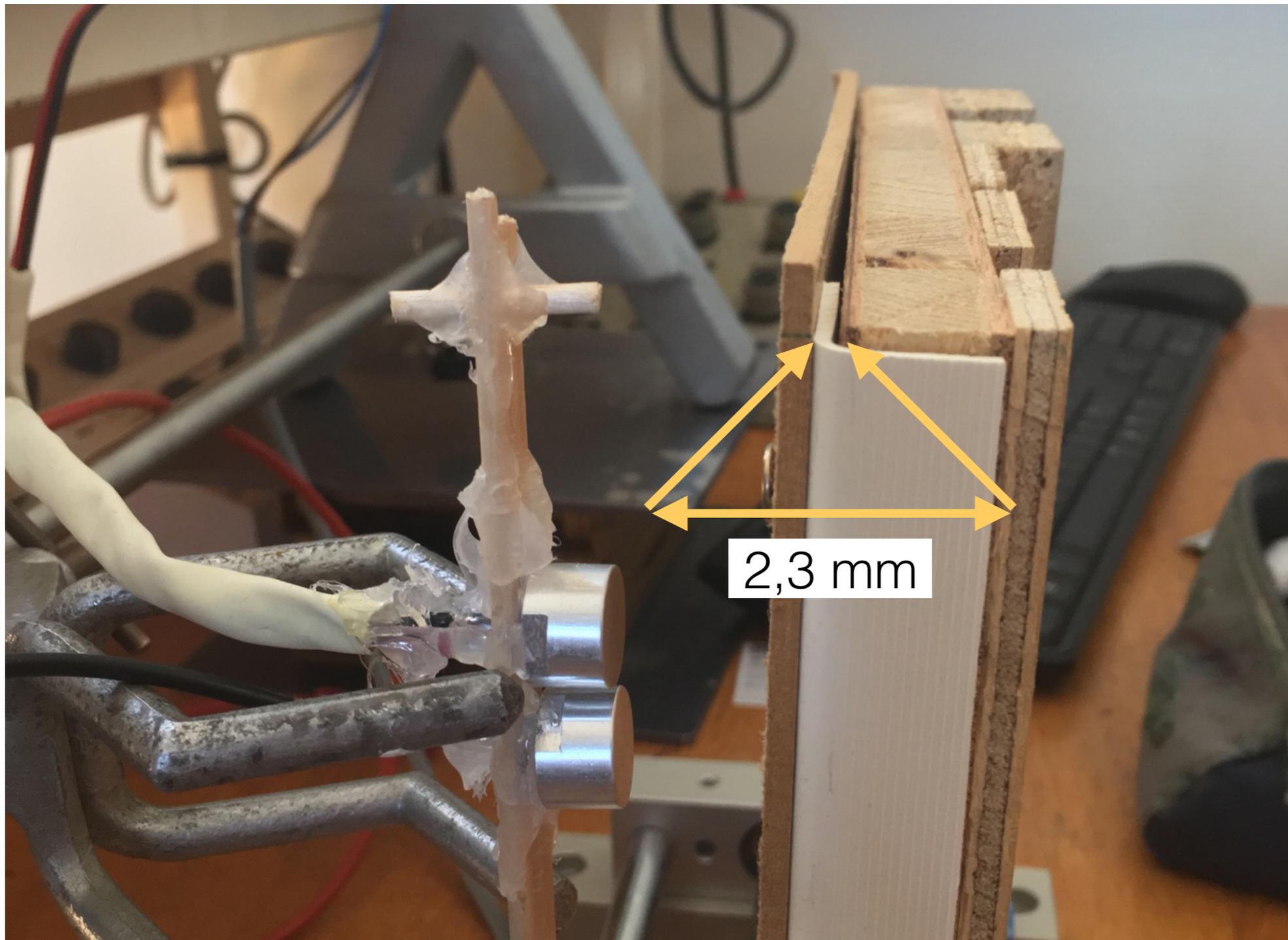
# Détermination de ses caractéristiques

Loi tension - vitesse du système



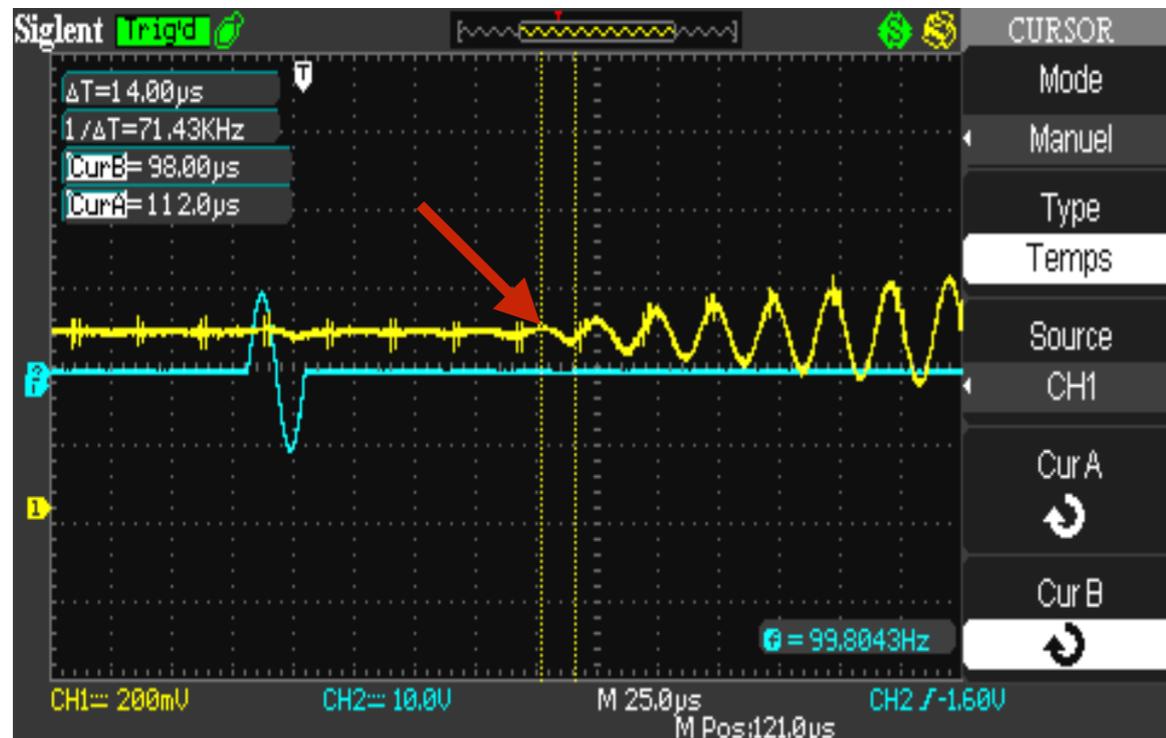
# Contrôle par pulse et temps de réponse

Planche test



# Contrôle par pulse et temps de réponse

Planche test

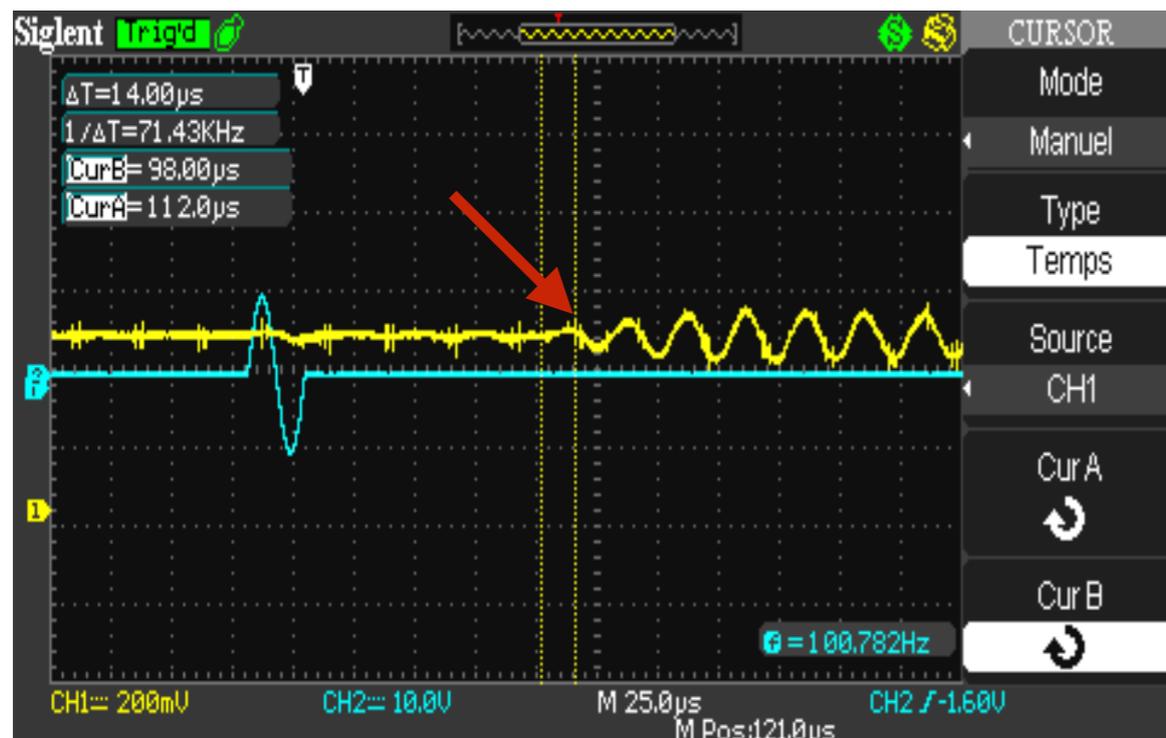


$$\Delta T_{\text{exp}} = (14 \pm 1) \mu s$$

$$\Delta L_{\text{exp}} = \frac{c \times \Delta T_{\text{exp}}}{2}$$

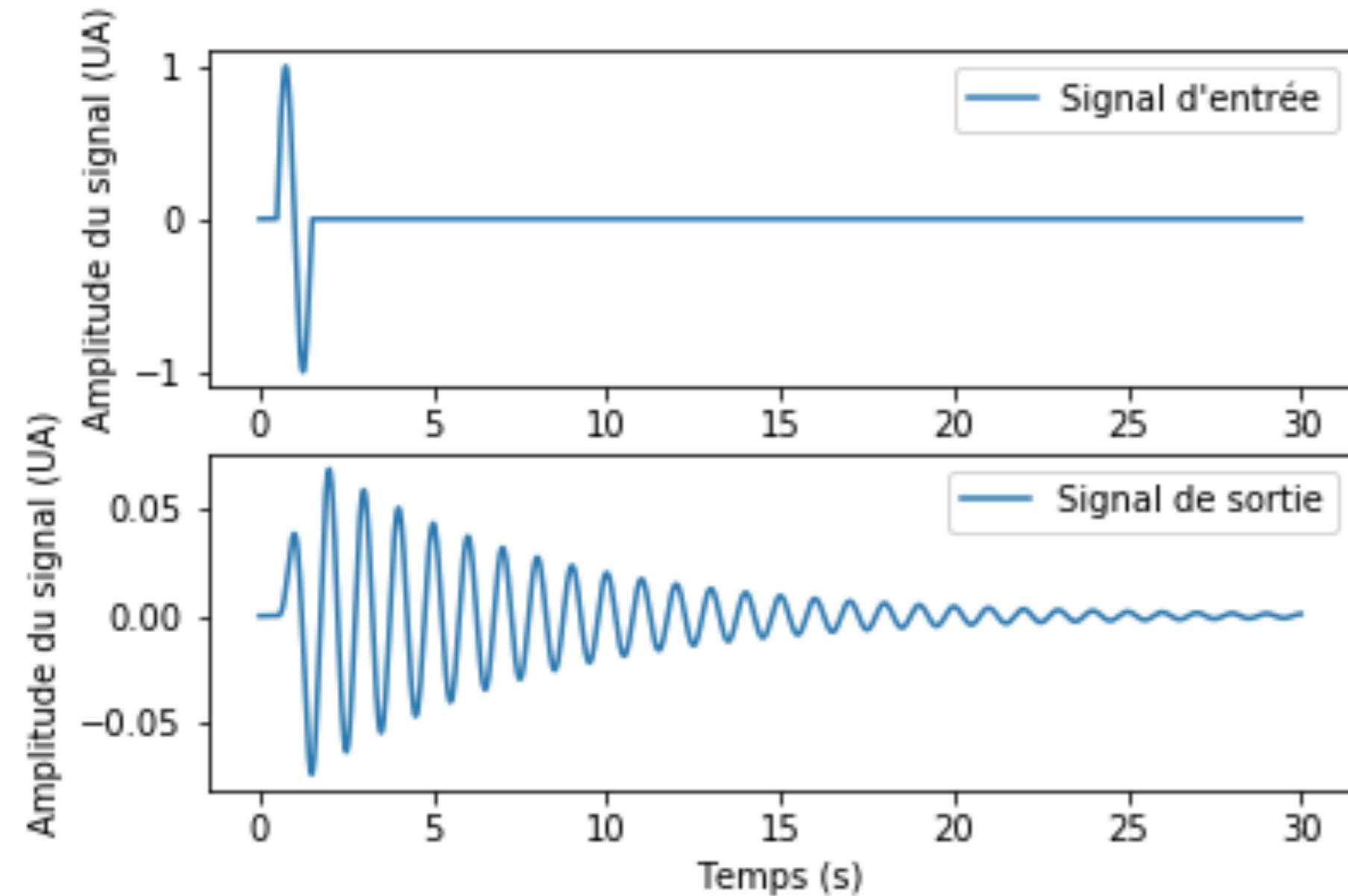
$$\Rightarrow \Delta L_{\text{exp}} = (2,4 \pm 0,2) \text{ mm}$$

$$\Delta L_{\text{réel}} = (2,3 \pm 0,1) \text{ mm}$$



# Contrôle par pulse et temps de réponse

Limites

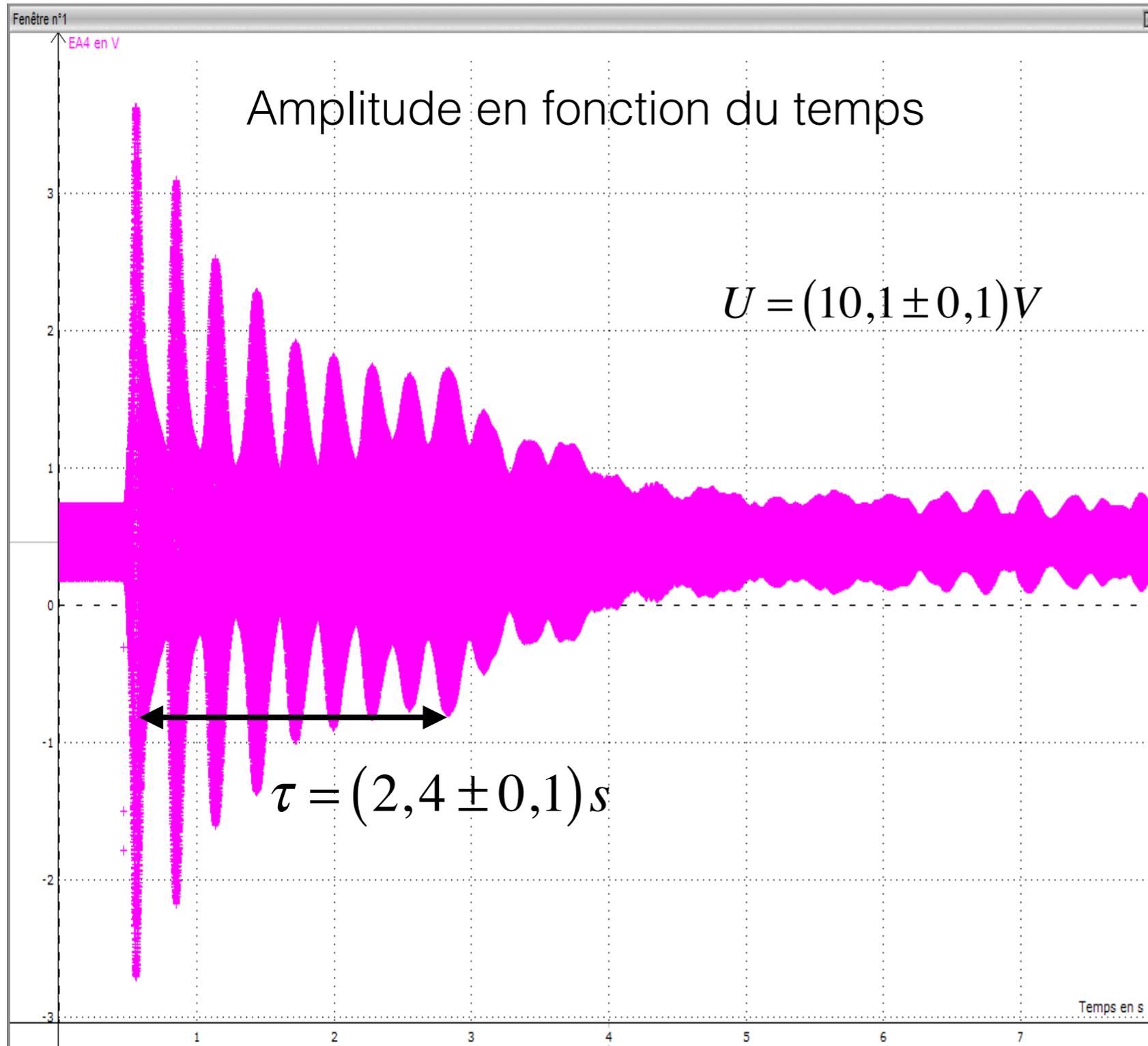


$$\begin{cases} \ddot{X} + \frac{\omega_0}{Q} \dot{X} + \omega_0 X = S(t) \\ Q \text{ grand} \end{cases}$$



Signal peu exploitable

## Mise en évidence des ondes stationnaires



Ondes stationnaires

$$d_{pic} = \frac{\tau}{8} \times v$$

$$d_{pic} = (3,8 \pm 2)mm$$

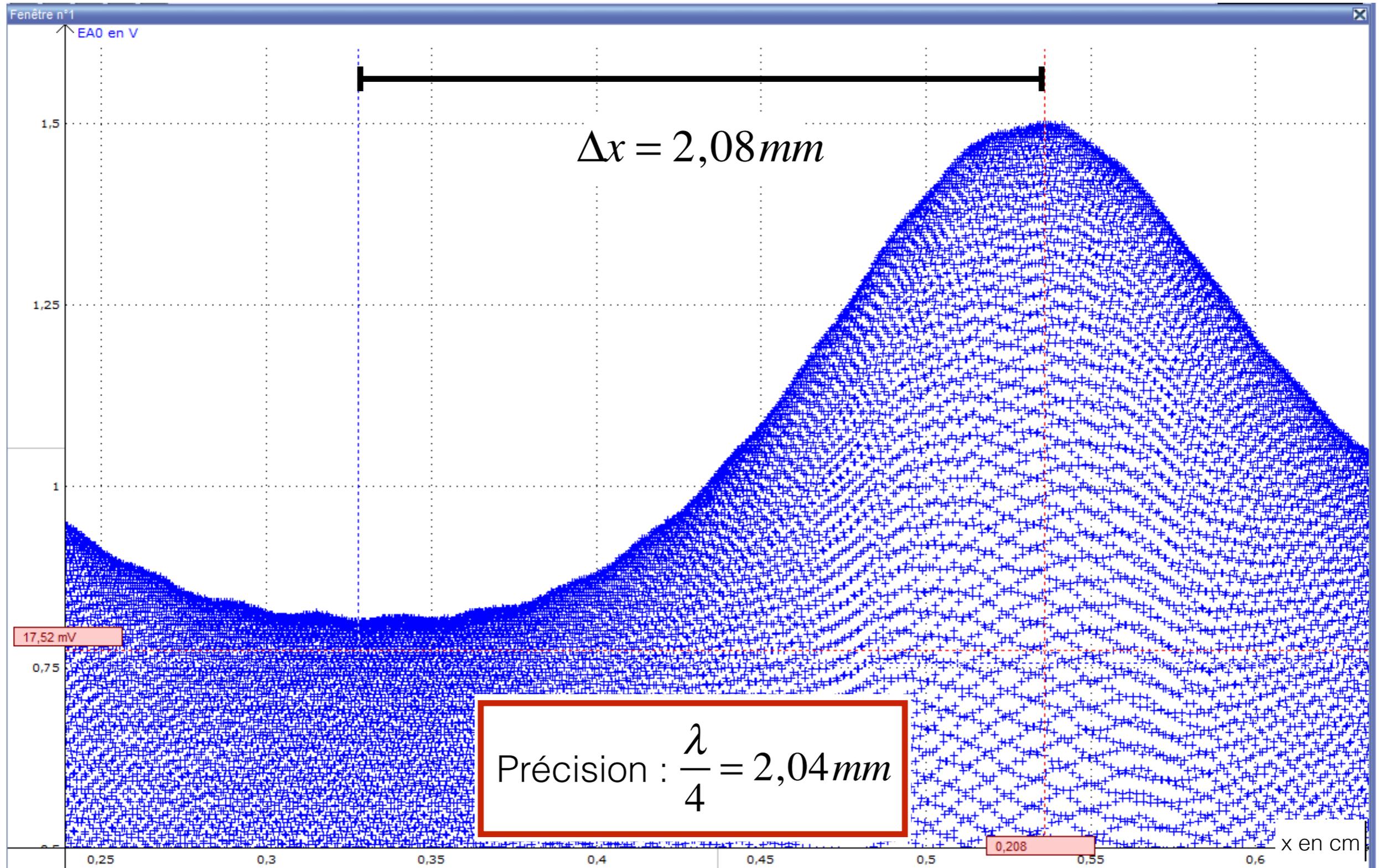
$$\frac{\lambda}{2} = 3,9mm$$

Distance de travail

$$d_{max} = \tau \times v$$

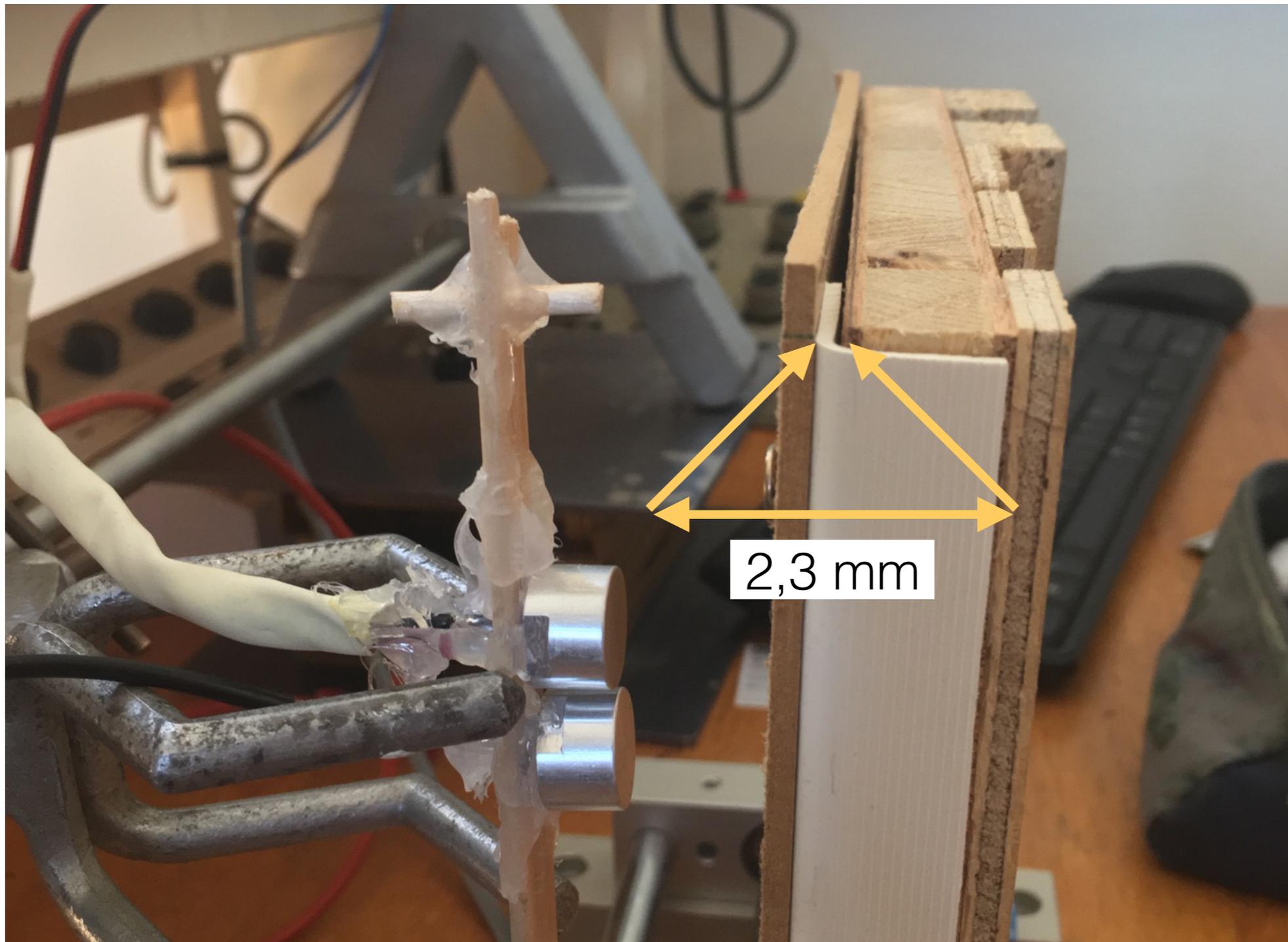
$$d_{max} = 3,0cm$$

## Contrôle par ondes stationnaires



# Contrôle par ondes stationnaires

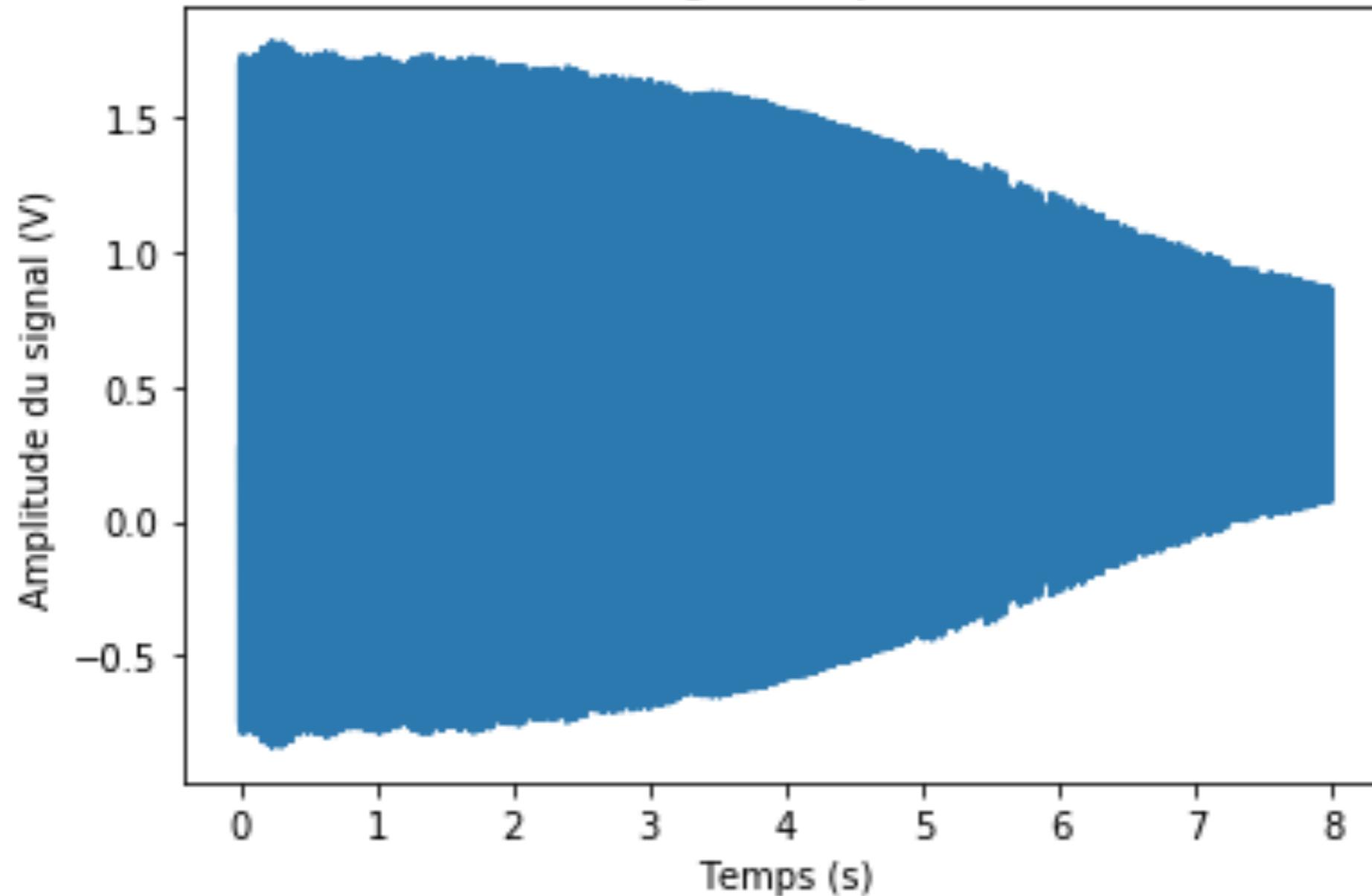
Planche test



# Contrôle par ondes stationnaires

Planche test

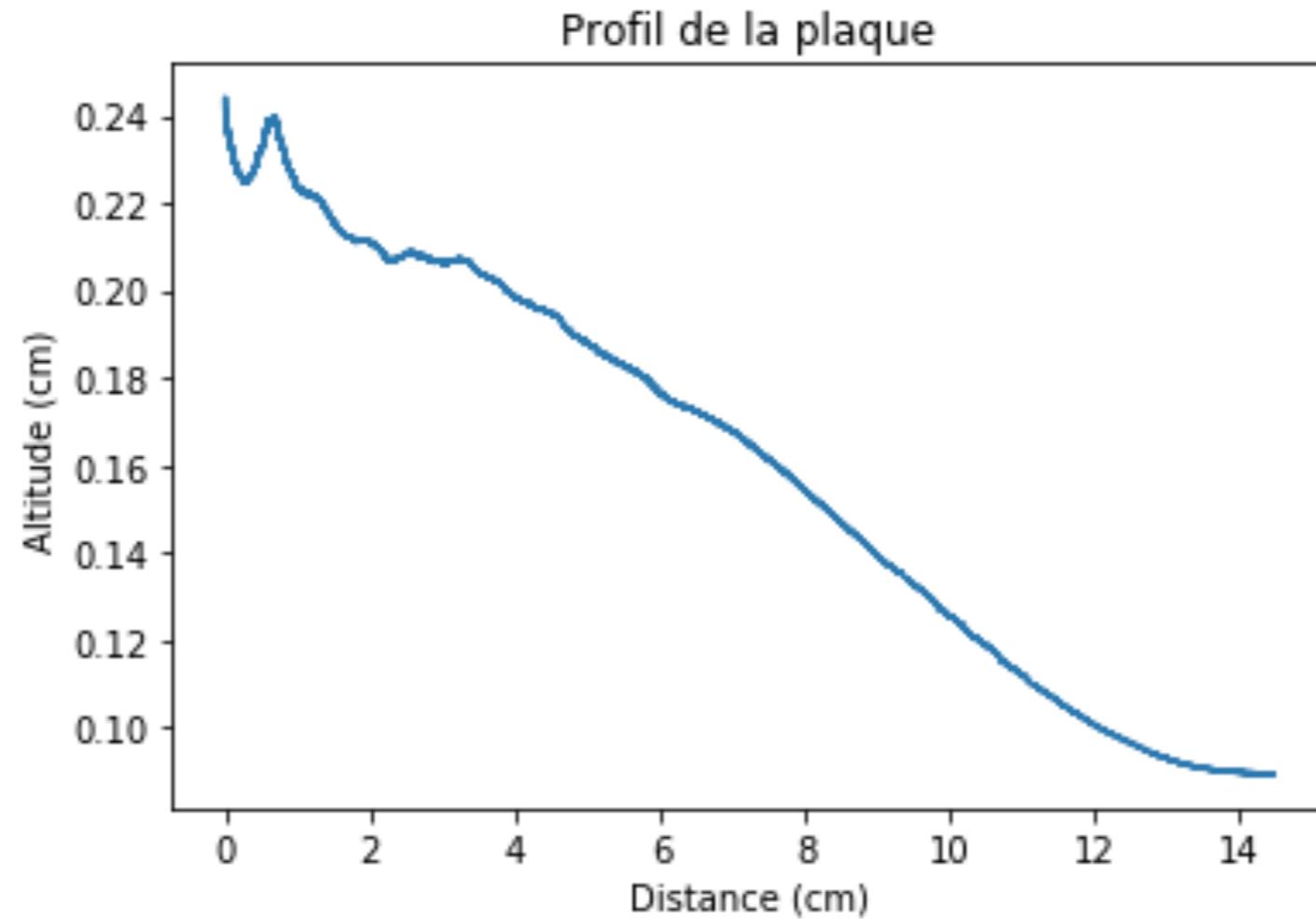
Signal acquis



Ventre placé sur la partie épaisse

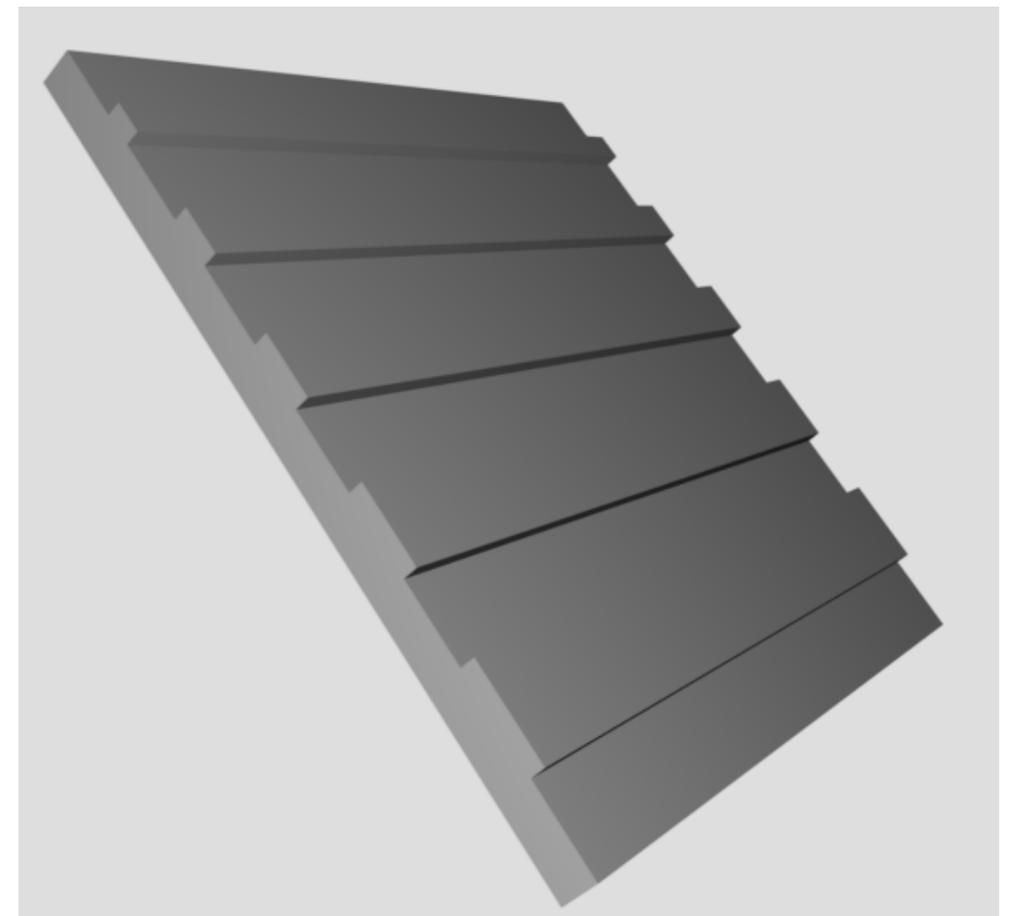
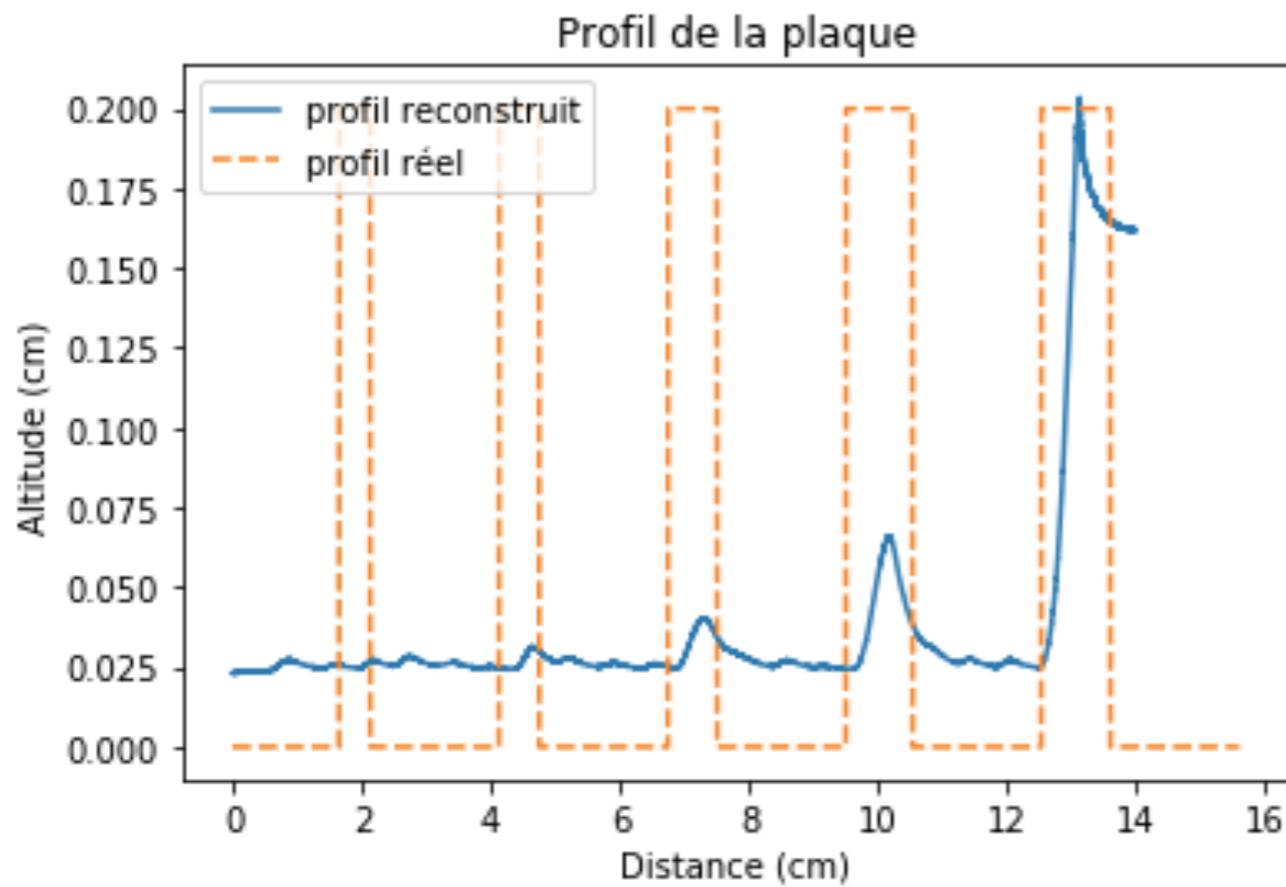
# Reconstruction de la surface

## Planche test



# Reconstruction de la surface

Plaque déformée et limites du système



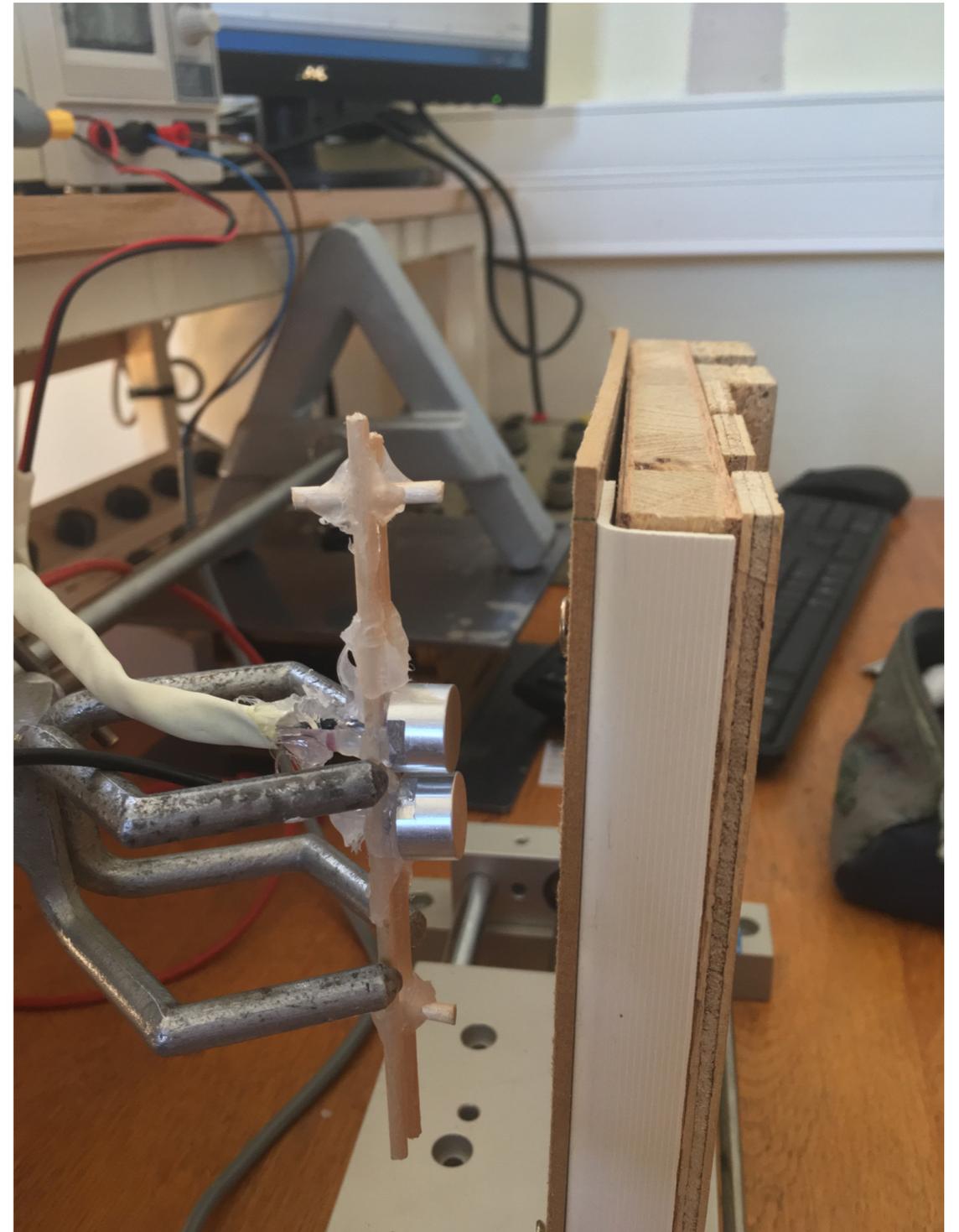
## Conclusions

Contrôle d'épaisseur : fiable

Contrôle de défauts : peu performant

Grande précision :  $\frac{\lambda}{4}$

Reconstruction du profil : correcte



# Annexes

## Code Python

```
def bout_sinusoid():
    signal = [0]
    temps = [0]
    T = 1
    w0 = 2*m.pi
    dt = 10**(-4)

    while temps[-1]<0.5:
        signal.append(0)
        temps.append( temps[-1]+dt )

    while 0.5<=temps[-1]<1.5:
        signal.append( m.sin( 2*m.pi*(temps[-1]-0.5) ) )
        temps.append( temps[-1]+dt )

    while 1.5<=temps[-1]<30:
        signal.append(0)
        temps.append( temps[-1]+dt )

    return(temps, signal)
```

```
def second_ordre(Q, signal):
    A = [0]
    V = [0]
    X = [0]
    dt = 10**(-4)
    i = 0
    w0 = 2*m.pi
    while len(X)<len(signal) :
        A.append( signal[i] - w0*V[i]/Q - w0**2*X[i] )
        V.append(V[i]+dt*A[i])
        X.append(X[i]+dt*V[i])
        i+=1
    return X
```

Simulation second ordre

# Annexes

## Code Python

```

def passe_bas(L, duree, freq_coup):
    N = len(L)
    dt = duree / N
    w0 = 2*m.pi*freq_coup
    absL = [abs(L[i]) for i in range(N)]
    filtreL = [ 2*max(L)/m.pi ]
    for n in range(1,N):
        filtreL.append( w0*dt*absL[n]/(1+w0*dt) + filtreL[n-1]/(1+w0*dt) )
    return filtreL

def surface(signal, longonde):
    profil = []
    Umax = max(signal)
    for k in range( len(signal) ):
        profil.append( longonde * np.arcsin( signal[k] / Umax ) / (2*m.pi) )
    return profil

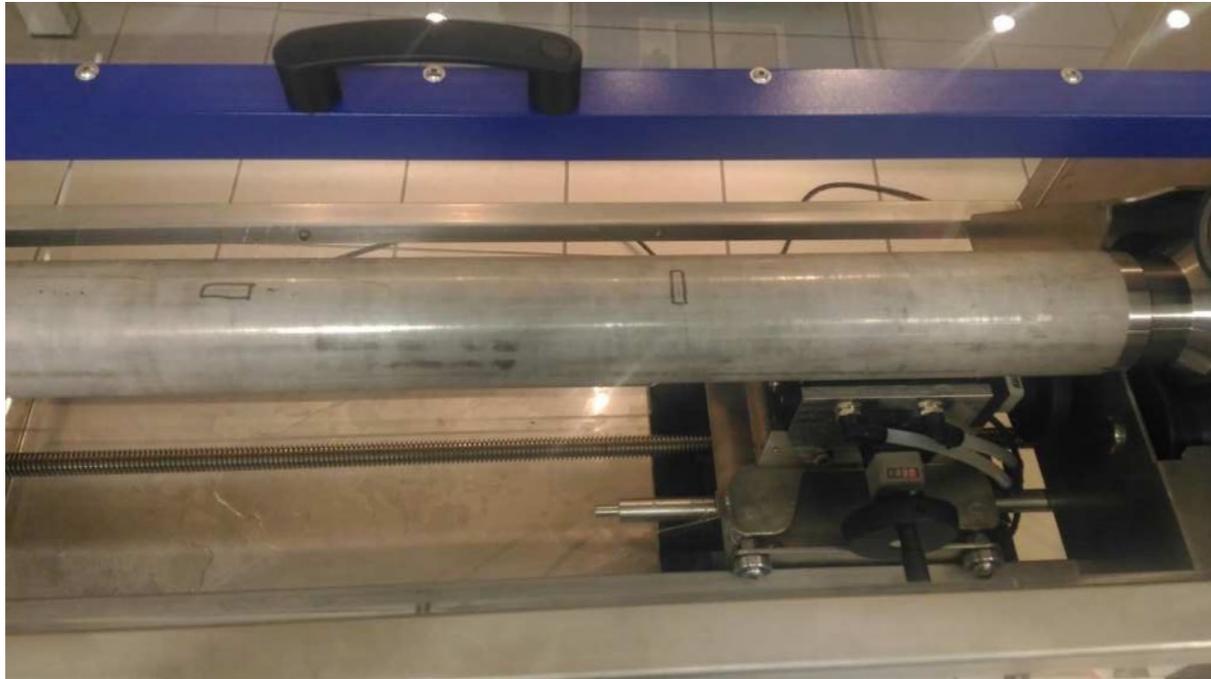
def temporel_vers_spatial(temps, tension):
    vitesse = .152*tension - .279
    spatial = [ T*vitesse for T in temps ]
    return spatial

```

Traitement du signal

# Annexes

## Photos de l'industrie



Systeme industriel de Montbard

