Development of a control and monitoring system for an ultra-short pulse autocorrelator.

Thesis submitted in partial fulfillment of the

Bachelor's degree in Engineering Physics

by

Quim Guerrero Casado

Under the supervision of

Crina Cojocaru i Jose Francisco Trull

ESCOLA TÈCNICA SUPERIOR D'ENGINYERIA DE TELECOMUNICACIÓ DE BARCELONA-UPC

UNIVERSITAT POLITÈCNICA DE CATALUNYA

JUNE 26, 2022

ACKNOWLEDGEMENTS

I would like to thank my supervisors, Jose and Crina for all their guidance and help. They are amazing researchers from whom I have learnt a great deal.

Quim Guerrero Casado

ABSTRACT

This project, entitled “Development of a control and monitoring system for an autocorrelator of ultrashort light pulses” is devoted to the study and implementation of single-shot characterization method of ultrashort laser pulses using second-harmonic generation in nonlinear ferroelectric crystals. The main objective is the development of a program that uses the experimental measurement of the autocorrelation trace and measures the duration and other parameters of the Gaussian pulse by image processing. The basic principle of operation is related to the second harmonic signal emitted by two crossing light pulses in a non-linear (NL) crystal, constituting the autocorrelation trace. As NL crystal, we use a ferroelectric crystal with random size and distribution of the NL domains, which allows the second harmonic generation in the transverse direction with respect to the beam propagation direction. This method had been implemented for pulses from a few tens to a few hundreds of femtoseconds. In this project, the method will be automatized and applied in the femtosecond regime. This technique presents different improvements against other traditional methods. Furthermore, it allows getting automatic phase matching without temperature control or angular alignment over a very wide spectrum.

The program will be initially built in the original setup (which works in the femtosecond regime), and it will consist of a Matlab program that reads the images taken by a camera and makes the calculations of the pulse duration, chirp parameter etc. and a LabVIEW program that allows to take the images and obtain the results in a user-friendly software. Once the program is complete, different experiments will be realized to observe how the program responds to different set-ups and lasers.

Contents

Chapter 1

Introduction

Nowadays, ultrashort laser pulses are becoming an important tool in numerous technological and scientific applications, such as materials processing, the measurements of the dynamics of complex molecules or optical communications. All these applications need a precise knowledge of the laser pulses properties, like the duration of the pulse, but a precise characterization of an ultrashort pulse properties is not easy to obtain since electronic measurements are limited to the ps scale.Shorter pulse duration, in the range of ps and fs domain, need top be measured by indirect methods. The pulses we will use in the experiments of this project, can be represented either in temporal or spectral domain, and since both domains are related by the Fourier transform, by only measuring the pulse intensity and phase in either temporal or spectral domain, we obtain the complete characterization of the pulse.

When working with pulses longer than 1 nanosecond, there are many electronic devices that measure different pulse properties. For example, an energy detector detects pulse energy, and a photodiode measures the pulse duration. Anyway, pulses with a duration lower than 100ps can not be measured by any electronic detector, which makes it impossible to make a direct measurement. The existing techniques of ultrashort pulse measurements usually rely on optical gating between the pulse and its replica that is typically realized through a nonlinear optical process, like the second harmonic generation [1].
The objective of this project is then, first, to analyse and understand the measurement methods for ultrashort laser pulses, implementing an image recording system on an already existing measurement system in the laboratory. And finally, build a program able to measure the pulse duration from the obtained measures and a visual environment that allows observing the signal, and analyse the obtained data

results in real time.
Before explaining this concepts in detail, a brief introduction to light pulses propagation in linear media must be taken.

1.1 Mathematical description of light pulses

The electric field of a light pulse can be expressed as a temporal-dependent function. Using the quasi-monochromatic approximation, it looks like:

$$\vec{E}(z,t) = \vec{E}_0^+(z,t)e^{-i\omega_0 t}$$

Where $\vec{E}_0^+(z,t)$ is known as the complex pulse envelope and ω_0 is the carrier frequency.
For the moment, we are assuming no transverse dependence of the light pulses, and we are not considering polarization effects (we can omit the vector character of the fields). Later in this chapter we will consider them both beside other additional effects. The expression of the pulse can also be written in terms of its real amplitude and temporal phase like:

$$E^+(t) = |E_0^+(t)|e^{i\phi(t)}e^{-i\omega_0 t} \tag{1.1a}$$

The real electric field is given by:

$$E(t) = Re[E^+(t)]$$

The temporal intensity of the pulse is proportional to the modulus squared of the real amplitude, and it can be used to describe the light pulse:

$$I(t) \propto |E_0^+|^2$$

$$U^+(t) = \sqrt{I(t)}e^{i\phi(t)}e^{-i\omega_0 t} \tag{1.1b}$$

We can also define these equations in the frequency space by using the Fourier transform:

$$\tilde{E}(\omega) = \int_{-\infty}^{+\infty} E(t)e^{i\omega t}dt \Leftrightarrow E(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \tilde{E}(\omega)e^{-i\omega t}d\omega \tag{1.2}$$

These mathematical expressions can be physically interpreted as the fact that a pulse is obtained as a superposition of different frequencies.
We can also factor out an oscillatory term proportional to the carrier frequency, define $\Omega = \omega - \omega_0$ and some other mathematical changes in order to obtain the spectral representation of 1.1a:

$$\tilde{E}^{+}(\omega_0 + \Omega) = |\tilde{E}_0^{+}(\Omega)|e^{i\Psi(\Omega)} \Rightarrow S(\Omega) \propto |\tilde{E}_0^{+}(\Omega)|^2 \tag{1.3}$$

Where $|\tilde{E}_0^{+}(\Omega)|$ is the spectral pulse envelope, $\Psi(\Omega)$ is the spectral phase, and $S(\Omega)$ is the pulse spectrum.

1.2 Pulse duration and bandwidth

The main goal of this project is to measure the temporal pulse duration. In general, pulse durations down to roughly 10 ps can be measured with the fastest available photodiodes in combination with fast sampling oscilloscopes. We will work with lasers with a Gaussian profile, which have different expressions to define the pulse duration or the spectral width. We can define full-width-half-maximum (FWHM), half-width-l/e (HW1/e), half-width-$1/e^2$ ($HW1/e^2$), root-mean-squared (RMS) width and equivalent pulse width. [2]
In this project, we will work with ($HW1/e^2$), defining T as the distance between the two points whose intensities are $1/e^2 = 0.135$ times the maximum value. Although we work with $HW1/e^2$, it is good to make some other explanations about FWHM. FWHM defines the pulse duration T as the time between the most- separated points that have half of the pulse's peak intensity.

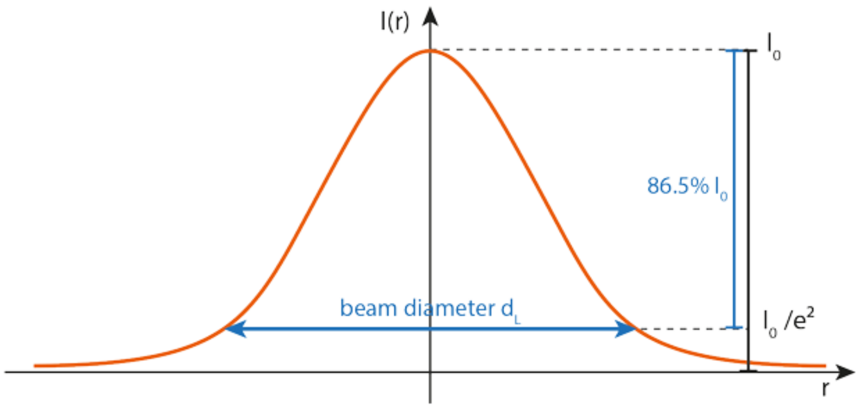


Figure 1.1: Definition of T working with $HW1/e^2$

In the same way, the spectral FWHM $\Delta\omega$ can be defined. And due to the fact that T and $\Delta\omega$ are related by Fourier transforms, they are dependent of each other. The product of temporal duration T and spectral width $\Delta\omega$ of a pulse is known as the Time-Bandwidth product, and it has a minimum value [3]:

$$\Delta\omega \cdot T = 2\pi \cdot \Delta\nu \cdot T \geq K \tag{1.4}$$

Where K is a constant that depends on the pulse temporal shape and can be derived analytically in each case. The smaller the TBP, the cleaner the pulse. The minimum possible value of the TBP comes when the spectral components are perfectly phase-locked, in other words, that is, when all the frequencies forming the pulse are added in phase. This case is named Fourier Transform Limited (FTL), and it represents the minimum pulse duration that can be attained for a given frequency spectrum.

For the phase-locked Gaussian pulse:

$$E(t) = exp(-(\frac{t}{T/4\sqrt{ln2}}))^2 \tag{1.5}$$

Which leads to:

$$S(\omega) = exp(-(\frac{\omega}{2\sqrt{ln2}/T}))^2 \tag{1.6}$$

where $\Delta\omega = (2\sqrt{ln2})^2/T$ and then:

$$TBP = \Delta\nu \cdot T = 0.441 = K \tag{1.7}$$

In a more general case, the different frequencies forming the pulse are not added in phase because the spectral phase is temporal-dependent. This leads to a broadening of the pulse, and consequently the time-bandwidth product is larger than from the FTL pulse.

1.3 Instantaneous frequency and chirp

Sometimes, in propagation of ultrashort pulses, it is relevant to make a general treatment with spatio-temporal coupling. In this project, spatial and temporal evolution of the fields can be decoupled, simplifying the problem. The expression for an optical pulse is then 1.1a, where we can omit the amplitude since we only care about the temporal pulse shape and duration. This expression can also be expressed as:

$$E_0^+(t) = e^{-i(\omega_0 t - \phi(t))} = e^{-i\Phi(t)}$$

The instantaneous frequency can then be defined as:

$$\omega_{inst}(t) = \frac{d\Phi(t)}{dt} = \omega_0 - \frac{d\phi(t)}{dt} \tag{1.8}$$

From this definition, one can see that if $\phi(t)$ is constant, ω_{inst} is constant across the pulse. When this happens, $\omega_{inst} = \omega_0$ which is the Fourier-transform-limited case. But, as explained before, in a more general case, $\phi(t)$ will be time-dependent. Using the Taylor expansion, it can be written like:

$$\phi(t) = \phi_0 + \phi_1 t + \frac{1}{2}\phi_2 t^2 + \frac{1}{3!}\phi_3 t^3 + ...$$

Introducing this expression in 1.8:

$$\omega_{inst}(t) = \omega_0 - \phi_1 - \phi_2 t - \frac{1}{2}\phi_3 t^2 - ...$$

Where only the first few terms are needed to describe the pulses we will work with. When the instantaneous frequency is time dependent, we can say that the pulse is chirped. The second order term ϕ_2 is called the linear chirp coefficient, and if it is the only temporal phase term ($\phi_2 >> \phi_3, \phi_4 ...$), we say that the pulse is linearly chirped. If the chirp coefficient is positive ($\phi_2 > 0$) the pulse is up-chirped and if it is negative ($\phi_2 < 0$) it is down-chirped. Any other higher terms of the temporal phase different from 0 lead to pulse distortions.

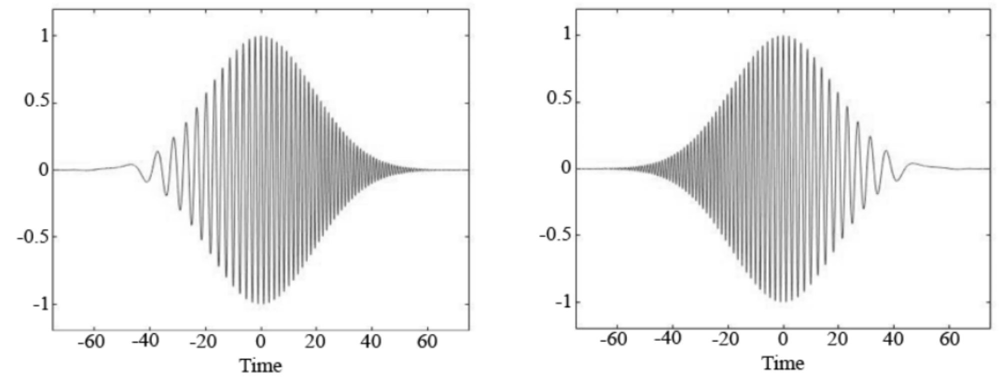


Figure 1.2: Up-chirped pulse on the left and down-chirped pulse on the right

1.4 Pulse Propagation in dispersive media

A dispersive medium is characterized by a frequency-dependent refractive index $n(\omega)$ and absorption coefficient. As it is known that any pulse is composed by a superposition of different frequencies added coherently. A change in refractive index also means a change in phase velocity, thus meaning that any different monochromatic wave composing the pulse will travel through the medium with different velocity and undergo different attenuations. Hence, the different spectral components will change their relative phases upon propagation. As a result, the pulse is broadened and chirped.

In this project, we will work with ultrashort pulses. The properties of ultrashort pulses can undergo complicated changes when propagating in transparent optical media, but if we consider an ultrashort pulse with low power propagation in a dispersive media, the only effect that must be taken into account is the chromatic dispersion.

This effect will be predominant in our experiment, since we will measure the chirp parameter evolution through our crystal. In order to describe this phenomenon, our goal will be to trace the evolution of the pulse with initial envelope $E^+(0,t)$ along the propagation distance z: $E^+(z,t)$. We consider that this pulse has a finite bandwidth $\Delta\omega$ centred at frequency ω_0. Since the ultrashort pulses we will be working with have durations of pico or femtoseconds, the relation $\Delta\omega << \omega_0$ holds, and we can define $\Omega = \omega - \omega_0$ like we did for 1.3. The chromatic dispersion can be defined via the Taylor expansion of the wave number k as a function of ω:

$$k(\omega) = \frac{n(\omega)\omega}{c} \tag{1.9a}$$

Using now the relation $\Delta\Omega << \omega_0$:

$$k(\omega) = k(\omega_0) + \left[\frac{\partial k}{\partial \omega}\right]_{\omega_0} (\omega - \omega_0) + \frac{1}{2}\left[\frac{\partial^2 k}{\partial \omega^2}\right]_{\omega_0} (\omega - \omega_0)^2 + ... \tag{1.9b}$$

Expressed in terms of Ω:

$$k(\Omega) = k(0) + \left[\frac{\partial k}{\partial \omega}\right]_{\omega_0} \Omega + \frac{1}{2}\left[\frac{\partial^2 k}{\partial \omega^2}\right]_{\omega_0} \Omega^2 + ...$$

$$= k(\omega_0) + \frac{\Omega}{u} + \frac{1}{2}\Omega^2 g + \frac{1}{6}\Omega^3 \beta_3 ... \tag{1.9c}$$

Where we defined some terms: the group velocity $u(\omega) = \left[\frac{\partial k}{\partial \omega}\right]^{-1}$, the group-velocity dispersion coefficient (GVD) $g(\omega) = \left[\frac{\partial^2 k}{\partial \omega^2}\right]$, and the third order dispersion (TOD) coefficient: $\beta_3(\omega) = \left[\frac{\partial^3 k}{\partial \omega^3}\right]$. Since the first two are the most important, let's describe them in more detail below:

1.4.1 Group velocity

The group velocity can also be defined as:

$$u(\omega) = \left[\frac{\partial k}{\partial \omega}\right]^{-1} = \left[\frac{\omega dn(\omega)/d\omega}{c} + \frac{n(\omega)}{c}\right]^{-1}$$

$$u(\omega) = \frac{c}{n(\omega) + \frac{dn(\omega)}{d\omega}\omega}$$

Which can be rewritten in terms of the wavelength using the relation:

$$\omega = \frac{2\pi c}{\lambda} \Rightarrow \frac{d\omega}{d\lambda} = -\frac{2\pi c}{\lambda^2} d\lambda = \frac{-\omega}{\lambda} d\lambda \Rightarrow \frac{\omega}{d\omega} = -\frac{\lambda}{d\lambda}$$

$$u(\lambda) = \frac{c}{n(\lambda) - \frac{dn(\lambda)}{d\lambda}\lambda} \tag{1.10}$$

1.4.2 Group-velocity dispersion

The group-velocity dispersion (GVD) is the phenomenon because of which the different frequency components from the pulse suffer different delays due to the fact that the group velocity itself is frequency dependent. As a result, the pulse spreads in time.

The GVD can be expressed in terms of the group velocity:

$$g(\omega) = \left[\frac{\partial^2 k}{\partial \omega^2}\right] = -\frac{1}{u(\omega^2)}\frac{du(\omega)}{d\omega}$$

We call normal group velocity dispersion, the regime in which the group velocity decreases with frequency, and anomalous group dispersion when the group velocity increases with frequency. In the first regime, low frequency components travel faster and in the second one high frequency components travel faster.

The GVD coefficient is a measure of the pulse-time broadening per unit distance per unit spectral width. And, in the same way as for the group velocity, it is usually expressed in terms of the wave length:

$$g(\lambda) = \frac{\lambda^3}{2\pi c^2}\left[\frac{d^2 n(\lambda}{d\lambda^2}\right] \tag{1.11}$$

1.4.3 Propagation equation

The general propagation equation can be quite complex. However, a simplified version of the propagation equation can be derived from the general wave equation by truncating the dispersion effects up to the second order:

$$\vec{\nabla} \times (\vec{\nabla} \times \vec{E}) + \mu_0 \frac{\partial^2 \vec{D}}{\partial t^2}$$

And the resulting simplified propagation equation is:

$$\frac{\partial A(t,z)}{\partial z} + \frac{ig}{2}\frac{\partial^2 A(t,z)}{\partial t^2} = 0 \tag{1.12}$$

Where A is the complex amplitude of the pulse, coming from:

$$E(t,z) = A(t,z)e^{i(k_0 z - \omega_0 t)} \tag{1.13}$$

The solution of this equation can be easily obtained, and is:

$$A(\omega, z) = A(\Omega + \omega_0, z) = A(\omega, 0)e^{i\frac{1}{2}g\Omega^2 z} \tag{1.14}$$

If TOD were also considered, the solution would be quite similar:

$$A(\omega, z) = A(\Omega + \omega_0, z) = A(\omega, 0)e^{i\frac{1}{2}g\Omega^2 z + i\frac{1}{6}\beta_3\Omega^3 z} \tag{1.15}$$

From these results, one can observe that $S(0, \omega) = S(z, \omega)$, or, in other words, that the spectrum remains constant during the pulse propagation. It is also important to see that when GVD dominates the dispersion the pulse acquires a quadratic phase, while if the TOD does dominate the dispersion, the pulse acquires a cubic phase. Finally, the electric field of the pulse in the temporal domain can be obtained straight forward using the Fourier transform:

$$E(t, z) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} E(\omega, z)e^{i\omega t}d\omega \tag{1.16}$$

1.5 Second harmonic generation and Phase-matching

The main goal of this project is to measure the duration of ultrashort pulses. This process is not direct, since today's oscilloscopes and other measurement instruments can deal and measure pulses up to the nanosecond's regime. When dealing with shorter pulses, it is necessary to use non-direct measurement methods. The existing techniques of ultrashort pulse measurements usually rely on optical gating between the pulse and its replica that is realized through generation of the second harmonics. [1] Noncollinear second-harmonic generation (SHG) is one of the best methods to measure ultrashort pulses. Two beams cross inside a nonlinear crystal with a small angle, and a second harmonic beam is generated in the forward direction if the phase matching conditions are fulfilled. Since these conditions are not trivial, but they must be fulfilled, let's make a brief explanation.

After some manipulations with the Maxwell equations in nonlinear media and doing some reasonable assumptions [4], one can obtain the equation of the SHG intensity:

$$I_2(L) = 2\epsilon_0 cn|E_2(L)|^2 = 2\sqrt{\frac{\mu_0}{\epsilon_2}}\omega_2^2 d_{eff}^2 |E_1^2|^2 L^2 \frac{sin^2(\Delta kL/2)}{(\Delta kL/2)^2} \tag{1.17}$$

Which allows us to obtain the conversion efficiency η as:

$$\eta(L) \equiv \frac{I_2(L)}{I_1} \propto \frac{d_{eff}^2 I_1 L^2}{n^3} \frac{sin^2(\Delta kL/2)}{(\Delta kL/2)^2} \tag{1.18}$$

Where $\epsilon_1 \approx \epsilon_2 \approx \epsilon_0 n^2$. This expression gives us the efficiency with which the original fundamental beam generates the second-harmonic. In this expression appears Δk which is the phase mismatch factor $\Delta k = K_2 - 2k_1$. When $\Delta k = 0$ the phase-matching condition is fulfilled, $sinc(\Delta kL/2) = 1$, and the conversion efficiency is proportional to some parameters and to the total intensity of the initial propagation beam. But $\Delta k = 0$ is not an easy condition to fulfil in the laboratory, and when $\Delta k \neq 0$ the conversion efficiency decrease drastically. This means that when the SH wave generated in a point z_1 arrives to a point z_2,it will not be in phase with the SH generated in z_2. This interference is described by the sinc function, which means that the efficiency and the intensity will both have an oscillatory behaviour. Furthermore, as the sinc function has a maximum at the phase π, the propagation distance that makes the accumulated phase difference π is known as the coherent length L_c, and when $L = L_c$ the nonlinear parametric process reverse its direction transferring energy back from SH to fundamental wave. From this distance, the intensity of the SH starts decreasing.

$$Lc \equiv \frac{\pi}{\Delta k} = \frac{\pi}{k_2 - 2k_1} = \frac{\lambda}{4|n_2 - n_1|} \tag{1.19}$$

Another possible definition from the coherence length is the maximum crystal length that is useful producing the SH power. From the figure below, one can observe how the intensity always decrease as Δk increases, and how the intensity has a maximum at the coherence length for a given Δk

As stated before, it is really complicated to reach the perfect phase matching condition, and it has been also shown that the efficiency decrease really fast if it is not reached. Anyway, there are some phase-matching techniques and also quasi

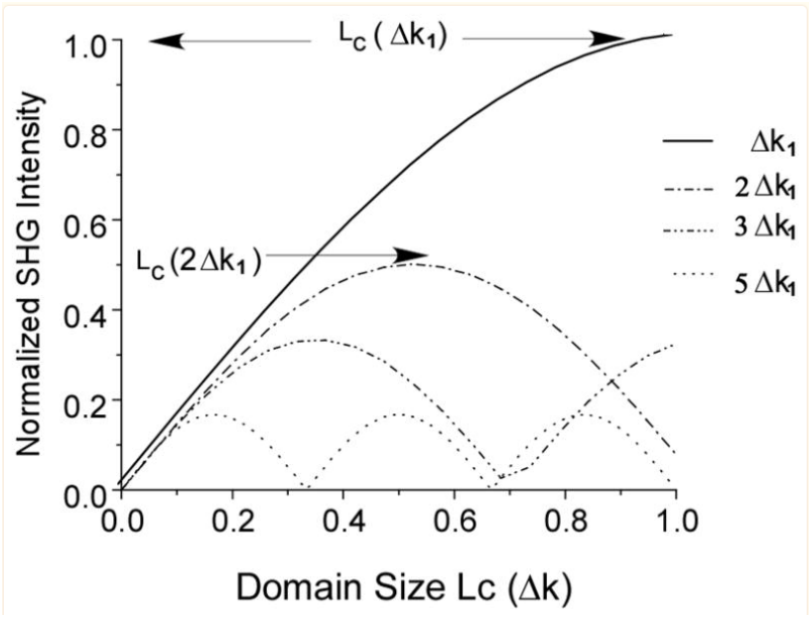


Figure 1.3: Normalized SHG intensity versus propagation distance for different Δk

phase-matching techniques that allow to work in proper conditions.

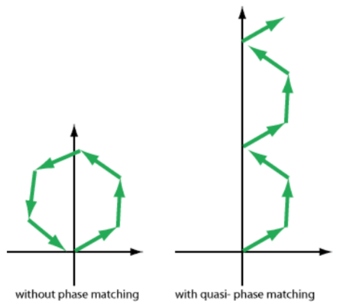
One of the most famous PM techniques is the birefringent phase-matching, which is based on the properties of the birefringent crystals. In those crystals, the electrons do not move in the same direction of the applied field, but they will follow a path imposed by the crystalline structure, thus resulting in a polarization vector that is not parallel to the electric field. Without entering any more detail, this technique is not useful for our thesis due to few reasons:

- PM condition is highly dependent on the incident angle, the polarization state of the beam and the wavelength. Since ultrashort pulses span over a broad frequency bandwidth, it is a big issue trying to achieve PM (or approximate PM) condition for all the frequencies.

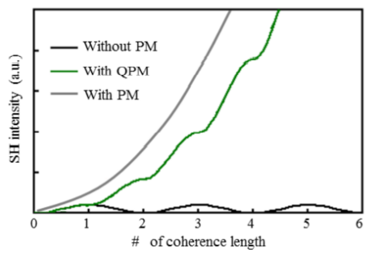
- The range of frequencies that can reach the PM condition (known as PM bandwidth) increases as the thickness of the crystal decreases, and since we will work with ultrashort pulses, very thin crystals will always be needed.

1.5.1 Quasi-phase-matching

Another important phase-matching technique is the one dimensional quasi-phase-matching (QPM) technique, which was experimentally proved in 1992 [5]. This technique is based on the oscillatory behaviour of the SHG intensity when the PM conditions are not satisfied, specially, in the first and biggest intensity peak at L_c. As stated before, when the propagation length is equal to L_c the phase difference is equal to π and the SHG reverses its direction transferring energy back to the fundamental beam. The QPM technique consists in adding periodically a phase shift of π, with that the intensity will keep increasing monotonically instead of decaying from L_c. In the figure below, it is illustrated the difference between the complex amplitude contributions from the nonlinear crystal to the SH wave when working with or without PM. It can be observed how with QPM the sign of the contributions is reversed every coherence length and the total amplitude keeps increasing. The realization process with which the π phase shift is applied consists



(a) Addition of amplitude contributions from different parts of the crystal



(b) SH intensity under different conditions.

in periodically invert the sign of second-order NL susceptibility $\chi^{(2)}$ of the material every coherent length. A strong field is applied to the ferroelectric nonlinear crystal for some time, so that the crystal orientation and thus the sign of the nonlinear coefficient are permanently reversed. The periodically spatial distribution creates a constant reciprocal lattice vector $G = 2\pi m/\Lambda$. With that, the phase-mismatch can be compensated through this G that applies directly in the momentum conservation relation as:

$$\vec{k}_2 - 2\vec{k}_1 = \Delta\vec{k} - \vec{G} = 0 \tag{1.20}$$

This technique has some benefits over the birefringent PM, for example, it can be implemented in non-birefringent crystals, thus solving the need for it to be thin. The problem is that this one dimensional quasi-phase-matching is still not useful to our experiments, because it still has the SHG bandwidth problem. The final QPM technique that solves this problem and consequently, the one we will use in the experiments of this thesis, is the QPM in 2D random ferroelectric crystals. Without entering much detail, in these crystals (such the Strontium Barium Niobate (SBN), which is the one used in this project) the linear susceptibility is constant, while the second order nonlinear susceptibility is spatially random modulated by the disordered ferroelectric domains [4]. The random $\chi^{(2)}$ distribution creates a continuous set of lattice vectors G, thus yielding t he SHG in the xy plane

1.6 Ultrashort pulse duration measurement techniques

Here, two different methods of measurement of ultrashort pulses using SHG will be explained. The first one is the most common one, and the second is a different faster method which we will use in this project.

1.6.1 Intensity autocorrelation method

The most common technique in ultrashort pulse measurement is the intensity autocorrelation (AC). This technique provides quantitative information about the temporal structure of an unknown signal. The unknown signal is split in two parts of equal intensity, and one of the pulses is delayed by τ. The two pulses are focussed into a nonlinear optical crystal, which is designed for efficient second harmonic generation over the full bandwidth of the pulse, what requires sufficient intensity and that the phase-matching conditions are fulfilled, either by crystal birefringence, QPM etc. With that, the two signals recombine inside the crystal and a SH signal is generated. This new SH signal is separated in the collinear and the noncollinear part thanks to the nonlinear crystal and its noncollinear geometry. The energy of this new signal is measured with the form $I_{AC}(\tau)$. By doing so, the same experiment needs to be repeated applying different delays and measuring the intensity of the new signal with an integrating detector, obtaining a relation between the delay τ

and the intensity of the noncollinear SH signal. The relation between this function and the original signal is given by:

$$I_{AC}(\tau) = \int_{-\infty}^{+\infty} |E(t)E(t-\tau)|^2 dt = \int_{-\infty}^{+\infty} I(t)I(t-\tau)dt \tag{1.21}$$

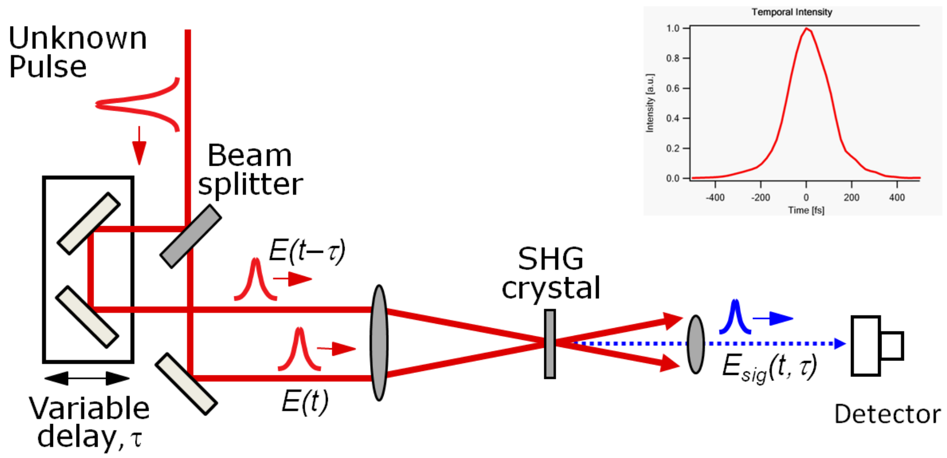


Figure 1.5: Intensity autocorrelator

The fact that the obtained function is symmetric means that this technique only provides information about the pulse duration, and although the intensity AC do not give information about the pulse shape, it depends on the assumed signal shape. Assuming a Gaussian-shaped signal, one obtains that the relation between the AC pulse and the original is $\sqrt{2}T = T_{AC}$.
Although this technique is the most used due to its simplicity, it has several limitations:

- It requires multiple measurements. This technique relies on measure the SHG for different delays in order to obtain the complete AC trace. This means that one has to repeat the same experiment many times, which makes this measurement process slow.

- Because the pulse duration can only be measured at the output of the crystal, one can never get rid of the pulse distortion acquired during the propagation inside the crystal.

- The PM conditions. In this case, it is highly dependent on the angle and wavelength, and as stated in 1, PM problems have to be fixed to obtain a proper efficiency. To do so, different nonlinear crystals are needed to work with pulses with different central wavelengths. Furthermore, thin crystals are also required in order to get a proper PM bandwidth, and the SHG efficiency is proportional to the square of crystal length, which makes it complicated to find a proper crystal length.

1.6.2 Transverse second-harmonic generation in SBN crystal

For these reasons, in this project we will use another autocorrelation technique, using a parametric conversion process in SBN crystals. These crystals consist of antiparallel nonlinear ferroelectric domains with random sizes and spatial distribution. In this media, the PM condition is relaxed and SHG has a similar efficiency over the full transparency range of the crystal, since the phase matching conditions are fulfilled over a broad frequency bandwidth. The thickness of the crystal is no longer a problem either, since the only limitation is the transparency region of the crystal.
Inside these crystals, depending on the direction of propagation of the original incident beam, the SHG can take the form of a plane (when the beam propagates perpendicular to the z axis) or a cone if it propagates in any other direction[6]. In this project, we will focus on the planar noncollinear SHG, that is emitted in the plane perpendicular to the two beams propagation direction, like can be observed in 1.6.

This planar SHG is formed by doing a similar procedure of the AC technique, when two replicas of the pulse overlap inside the SBN crystal, and it represents the autocorrelation of the pulse. Different pulse overlap regions are directly formed at different positions of the planar SHG plane, thus meaning that in a single shot one can not only obtain the pulse duration and the initial chirp parameter of the original pulse, but also the evolution of the interaction between the two replicas of the beam inside the crystal.
One big difference between this technique and the commercial AC, is that the SHG signal is captured at 90° with respect to the propagation direction of the incident beams.

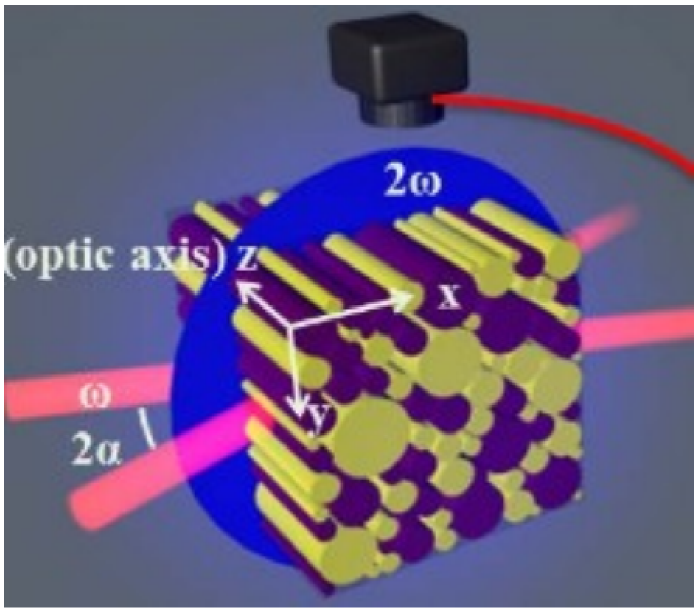


Figure 1.6: Representation of the transverse AC Setup [6]

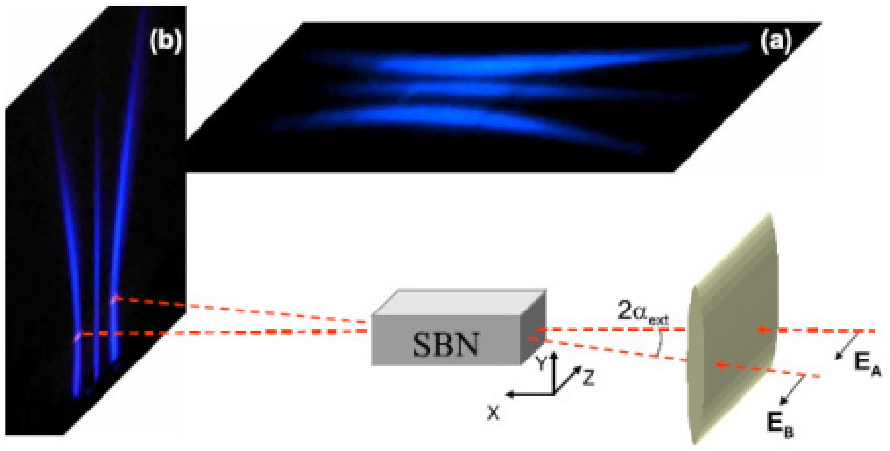


Figure 1.7: Planar SHG in SBN crystal [6]

The experimental setup is quite similar to the AC technique. The fundamental beam is split in two delayed replicas, that enter the SBN crystal with angles $-\alpha$ and α to the x-axis (as shown in 1.7). Each one of the beams generates a SHG in the form of a cone, and the interaction between the two beams inside the crystal gives rise to the above-mentioned planar noncollinear SH emission [7]. This emission can not only be measured in the transverse direction, but also in the forward direction, but it is easier to measure it from above because the desired signal is more isolated from the two other cone-shape emissions. The SH signal can be then detected by a CCD camera placed above the SBN crystal, imaging the xz-plane 1.7, and recording the evolution of the SH signal through the crystal. We assume that the two incident

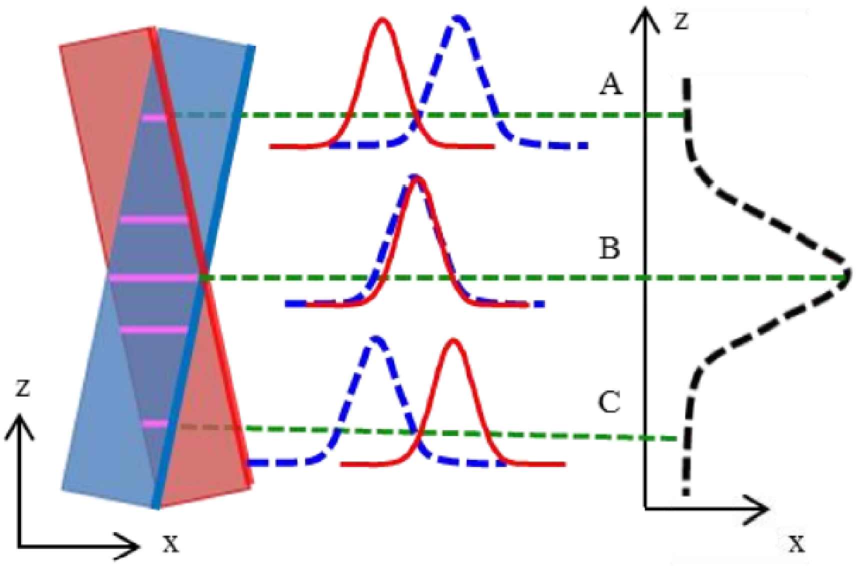


Figure 1.8: Interaction of two pulse replicas in SBN crystal [4]

beams have a Gaussian spatial and temporal profile, and that they cross at an angle 2α at the entrance of the crystal. Their amplitudes would be:

$$A_1 = A_{10}exp(-\frac{z_1^2}{2\rho^2})exp(\frac{-(t-x_1/u)^2}{2T^2}), \tag{1.22a}$$

$$A_2 = A_{20}exp(-\frac{z_2^2}{2\rho^2})exp(\frac{-(t-x_2/u)^2}{2T^2}), \tag{1.22b}$$

Where (x_1, z_1) and (x_2, z_2) are the coordinates in reference systems oriented along the propagation direction of each one of the beams 1.8, $u = c/n$ is the light speed inside the crystal, and ρ and T are half beam and pulse width at $1/e$ levels in intensity, respectively.

If we assume that the random QPM process fulfil the PM conditions [1], and introducing a common coordinate system for both beams, we obtain:

$$B(x, y, z) \propto P^{(2)}(2\omega) \propto d_{eff}^{(2)} A_1 A_2 \tag{1.23}$$

Now, assuming that the SH signal recorded by the camera is actually a cross-correlation function of the two incident pulses [7], the SH field generated by the two Gaussian pulses is represented as:

$$B(x,z,t) = B_0(x)exp(-\frac{z^2cos^2\alpha + x^2sin^2\alpha}{\rho^2}) \times exp(-\frac{(tu - xcos\alpha)^2 + z^2sin^2\alpha}{u^2T_c^2(x)}) \tag{1.24}$$

Where α is the halph angle between the wave vectors of the two intersecting fundamental beams inside the SBN and B_0 is an amplitude.
The generated signal emitted from the overlapping region of the fundamental beams, as can be observed in 1.7 and 1.8. This region depends on the spatial extent of the beam and the temporal duration of the pulses. When working with long pulses $T >> 2\rho tan\alpha/u$, the region is limited by the spatial dimensions of the beams, while when working with short pulses $T << 2\rho tan\alpha/u$ it is limited by the pulse lengths.
The resulting intensity of the SH generated is give by:

$$I_{2\omega}(x,z) = \frac{T^4 I_{2\omega}(0)}{T_{ch}^4(x+L/2)}exp(-\frac{2z^2sin^2\alpha}{u^2T_{ch}^2(x+L/2)}) \times exp(-\frac{2z^2cos^2\alpha + 2x^2sin^2\alpha}{\rho^2}) \tag{1.25}$$

where

$$T_{ch}(x) = T\left[\left(1+\frac{C\beta_2 x}{T^2}\right)^2 + \left(\frac{\beta_2 x}{T^2}\right)\right]^{1/2} \tag{1.26}$$

When a time delay between the two pulses is applied, the SH appears moving along the Z-direction, and its thickness depends directly on the pulse duration.

$$\Delta z(x) = \frac{uT_{ch}(x)}{\sqrt{2}sin\alpha} \tag{1.27}$$

This thickness is the most important unit in this project, as it is the only one that is measured directly from the obtained image of the SH trace. One example of the images we have worked with in the laboratory is in 1.9. This $\Delta(z)(x)$ is directly obtained measuring the thickness of the autocorrelation trace obtained directly from a picture taken in the laboratory.

What will be very important in this work, because that implies that using a slow enough pulse, a bigger crystal will be needed.

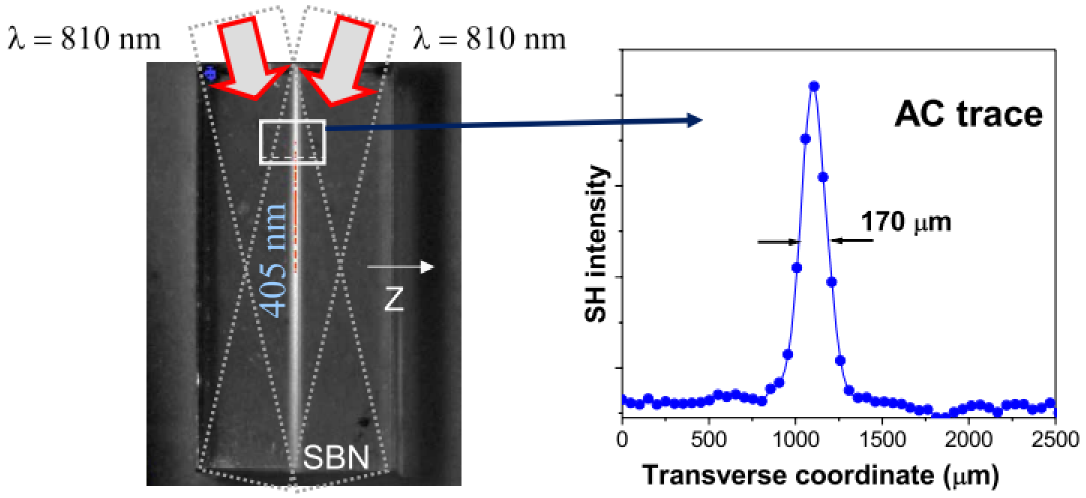


Figure 1.9: The AC trace of transversely emitted SHG from two intersecting fundamental beams [6]

If we measure the width of the SH trace at the input of the crystal $\Delta z(0)$, we obtain a measure of the time duration:

$$T = (\sqrt{2} sin\alpha)\Delta z(0)/u \tag{1.28}$$

Which is one of the most important values we will be searching for in this project. The same equation can be obtained working with the experimentally measurable FWHM τ

$$\tau = (\sqrt{2} sin\alpha_{ext})\Delta z_{FWHM}(0)/c \tag{1.29}$$

Where this α_{ext} is the one observed in 1.7.
An equation that gives us the initial chirp of the pulse can be derived using 1.26 and 1.27. When β_2 is known, we only need to measure $\Delta z(x)$ at a given x to obtain the initial chirp of the pulse:

$$C = \frac{\sqrt{(4\Delta z(0)^2 \Delta z(x)^2/u^4) sin^4\alpha - (x\beta_2)^2}}{x\beta_2} - \frac{(2\Delta z(0)^2/u^2)\, sin^2\alpha}{x\beta_2} \tag{1.30}$$

For β_2 and C having the same sign, the SH trace will grow constantly with x, the distance from the entrance of the crystal, while when they have different signs, the trace will start decreasing its width, since it find its minimum, from which the width begins to grow.

Chapter 2

Automation programs

The experimental methods explained in the previous section had been studied previously. The set-up corresponding to this technique has been already implemented in the laboratory, shown in 2.1. Anyway, this set-up was not automized, it

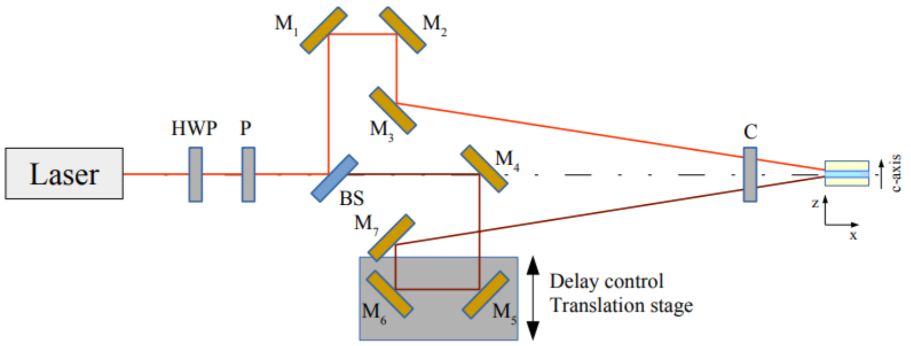


Figure 2.1: Experiment set-up

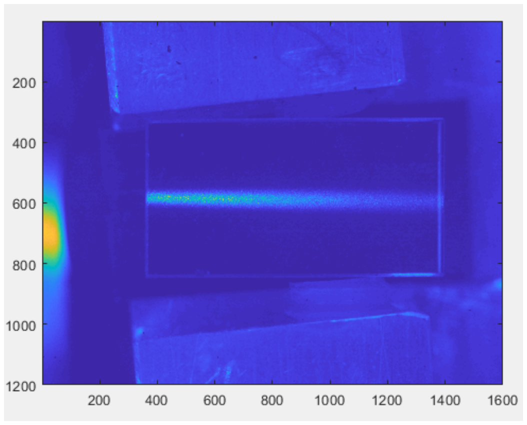
only allowed to take one single picture at a time to obtain the pulse duration after using the equations from 1.27 to 1.30. The key goal of this project is to build up a program that allows to automate this measurement process and obtain the desired parameters (from 1.27 to 1.30) and different graphs in real time. This is something that have never been realized before, and it would mean a major improvement, because it would make the measurement process much faster and easy to do.
The program consists of two main different parts, one programmed using Matlab, and the other using LabVIEW.
The Matlab program is in charge of the mathematics and the calculations. Different versions have been proved, but essentially, it takes an image of the SH trace as an input, and it gives as an output the duration of the incident pulse, the initial chirp

parameter, the evolution of the beam radius over the crystal etc.
The LabVIEW part, is in charge of the automation. It is the part that the user of the program will see and interact with. It shows the images recorded by the camera in real time, and allows the user to take the photos and see the desired results.

2.1 Matlab program

The famous and well-known Matlab is a non-free software and programming language and numeric computing environment developed by Mathworks. It has been the chosen programming language above others because it is very simple and fast to import external data (like images) as it has native toolboxes designed for image processing, and it is really fast on doing the mathematical calculations. Anyway, Python could have also been used, but it would have been slower. The worst aspect about Matlab would be the fact that it is non-free, but since the UPC provides licences to its students and professors, it has not been an inconvenient. Anyhow, this code could be translated to Python in order to obtain a free-license version of the program.
Now, the built program will be explained in detail. Firstly, the images are imported to the script. The photos are red in a ".txt" file, what means that Matlab transforms it directly to a matrix which can be easily manipulated from now on. We tried two different cameras from the laboratory. The first camera we tried is the ueye camera UI-1480SE-M-HQ, which we discarded after taking some images and processing them. This camera only took the photos in image formats like "JPG" and although Matlab is able to transform image formats to a matrix in such a way it does with the "txt" files, the resolution becomes worse respect the other case. This is the reason why we finally we opted for the *Spiricon SP620U*, because it captures the image with a proper resolution in a ".csv" file that can be instantly changed to ".txt". Another plus point for this camera, is that it allows to be controlled from LabVIEW using its packages.

One example of an image obtained by this camera is 2.2. The SH signal we want to measure is the orange-yellow line from the middle, thus meaning we want to select the region of interest (ROI) of the image. The light blue part surrounding the yellow line is the light emitted by the two cones from each of the two incident split



(a) Image obtained with Spiricon SP620U

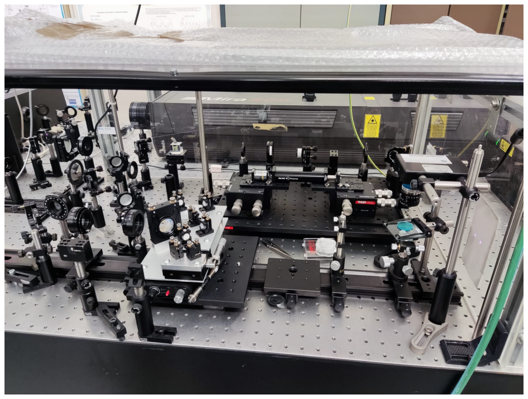
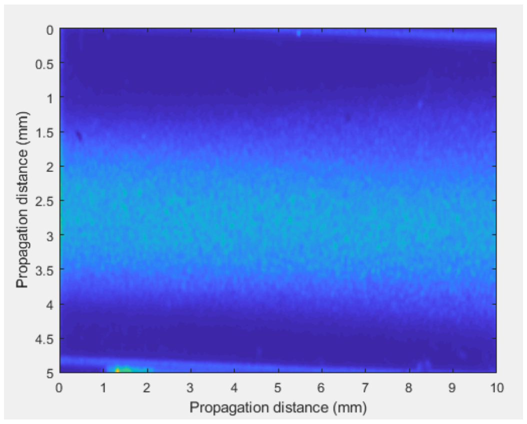

(b) Experimental set-up

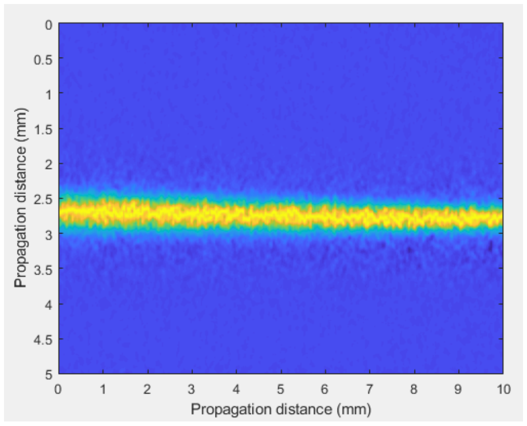
Figure 2.2: At the left, an image obtained by the camera. At the right, the experimental set-up in the laboratory.

beams, and there are other non-interesting parts in the image. In order to retrieve the important part of the image, we chose the method of taking a background image. This is easy to do in the laboratory, where one can move manually the position of the beams to obtain an image like 2.3a.
With that, Matlab reads 2 images, one from the SH signal itself, like 2.2a and another one of the background only (which contains the light from the two cones described in the previous section) and it subtracts the Background matrix from the signal matrix. Although there are other methods of retrieving the important part of the signal, we chose this one because it was not only easy to apply in the program but also for the user, who can decide with which background and set up wants to work, and change the background when needed.

It is still necessary to select the important part of the image, and to do so, no methods of image recognition worked properly, since the photos are taken in the dark in the laboratory and it is so difficult to distinguish what is the crystal or not, and since to capture the whole crystal including the non illuminated by the pulses part was necessary, the better option was to fix the camera in the set-up, and always cut the same part of the image.
After that, the resulting matrix is treated in order to obtain a cleaner one with less distortion. Different functions were tested. The first option was to interpolate the matrix to obtain a new one with more points and therefore more resolution, but

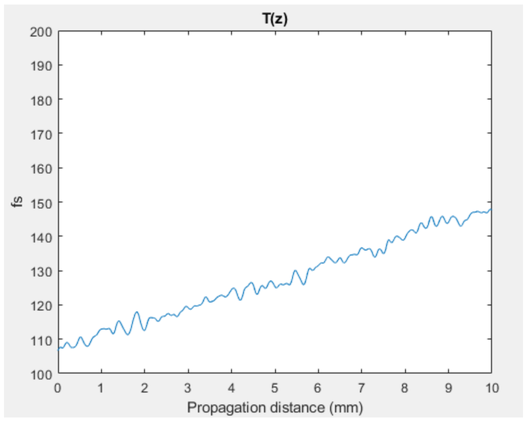


(a) Image of the background only



(b) Image without the background

the obtained results make not significant changes, and, additionally, the program becomes much slower. It was impossible to distinguish the two images before and after interpolate, and although in the resulting graphics of, for example, T(z) were a bit cleaner, the computational cost was not worth the implementation.
With these bad results, the next approach was trying using image processing toolboxes. One can observe how the graph oscillated sharply, so the idea was to smooth the obtained T(z) function using moving average, and the obtained solution was much cleaner.



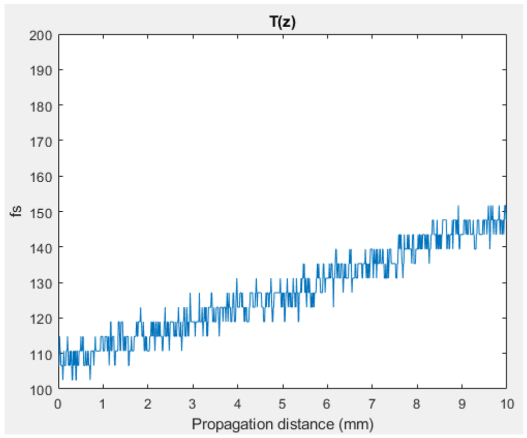


Figure 2.4: Duration of the pulse (T) along the crystal length (z). On the left side, the result using moving average, and on the right side, without interpolation.

What the Matlab function did was just:

$$ynew(1) = y(1);$$
$$ynew(2) = \frac{y(1) + y(2) + y(3)}{3};$$

And from there:

$$ynew(i) = \frac{y(i-2) + y(i-1) + y(i) + y(i+1) + y(i+2)}{5}$$

The last image processing done was a Gaussian filter applied to the image matrix, because we are working with Gaussian profiles. This is because the obtained Gaussian curves (the different pulses along the z-axis) were still too sharp after applying the moving average method.
Once the image processing part is over, the program starts making the calculations. The most important part is the calculation of the beam radius using a numerical knife-edge method. This method consists in a very sharp edge placed at the z plane where the beam is measured. A detector placed on the beam axis measures the beam energy, and as the edge moves into the beam, the energy going into the detector decreases. As a result, the beam radius corresponds to the edge displacement between the positions for which the intensity measured corresponds to the 16% and the 84% of the total beam intensity. The objective of this method applied numerically corresponds then to calculate the D_{16-84} profile, which is the equivalent to $\Delta z(x)$ from 1.27. By doing so, the beam width evolution across the crystal is obtained. The program calculates the points where the area of the Gaussian are the 16% and the 84% of the total area, respectively, and it calculates the difference. Different numerical integration methods have been tested, but finally, the best option was to use the trapezoidal method, which is fast and precise for sinusoidal-type functions like the Gaussian profiles we are working with. With that, using 1.28 one can calculate the desired pulse duration. In any case, the obtained values are measured in pixels, so it is necessary to change the units to the international system to obtain Δz in mm and T in fs.

One of the most important points in the technique is to obtain the right calibration factor to relate the pixel size with trace size and to femtoseconds. Different methods could be used. The first method applied to change the units of our system was by using an autocorrelation technique. By a thorough work in the laboratory, different measurements of the SH trace separated from them by a given interval

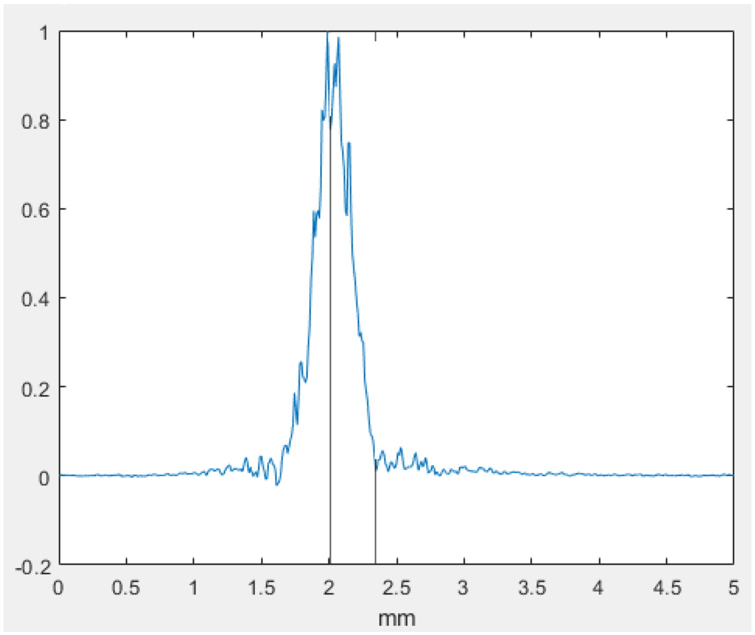


Figure 2.5: Gaussian profile of the laser, with lines at the points where the area is the 16% and the 84% of the total area. The program calculates the difference between these 2 points as Δz

were realized. With that, one obtain different images of the SH trace displaced one from the other, and by measuring the distance between measurements in the laboratory precisely, one can compare the images in Matlab and correlate the distance measured in the lab with the distance in pixels between the images. To do so, there are some mathematics to be used. When a little displacement is applied between the two incident split beams, one can know the applied delay on time by simply applying $\delta t = \delta x/c$. Changing the pulse delay by a known amount δt, induces displacement of the correlation trace in the Z-direction $\delta z = c\delta t/(2 sin\alpha)$, which can be precisely measured [1]. In 2.6 the distance between the two traces in the lab was $\delta x = 0.12mm$, which corresponds to $\delta z = 0.68842279mm$. Counting in the image, one can see that the distance between the two traces is 48 pixels, what let us a relation of $0.01434mm/píxel$.

The second method is much simpler, it consists in take the total amount of pixels from the crystal, and take into account that the crystal used for this project measurements are 10x5x5mm, what means that one needs to multiply the obtained values of Δz by 5mm and divide them by the number of pixels on that direction of the crystal. For example, in 2.6 the total number of pixels is 341, what gives the

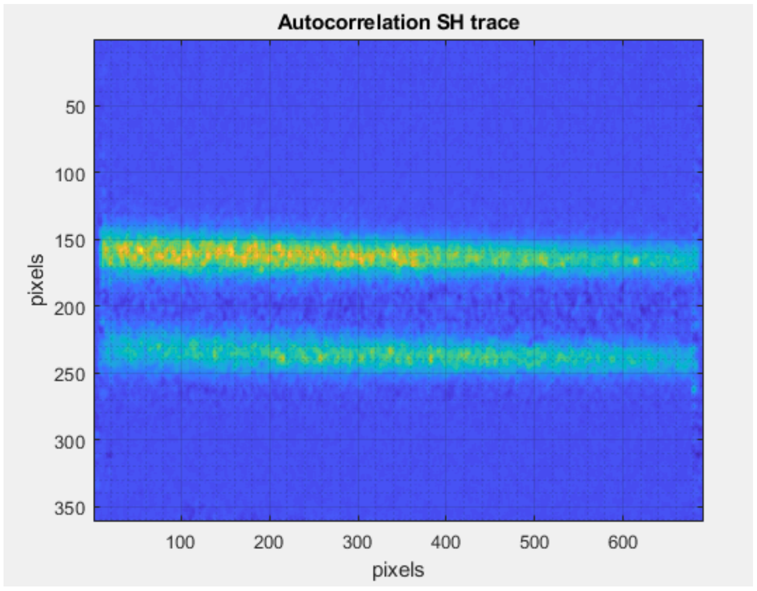


Figure 2.6: Two autocorrelated signals added up in the same image.

result of $5/341 = 0.01466mm/pixel$. One can observe how using both methods one obtain a very similar result, but since this second method is much faster to apply as it does not need to take images of different displaced traces, we chose to use this second easier method.
With the values of Δz and T measured, one way to check if these results make sense, is to plot a Gaussian profile from the processed image matrix and compare it to a theoretical Gaussian calculated by using the beam radius obtained: $I_0 e^{-\frac{2x^2}{D^2_{16-84}}}$ and taking x_o (the centre of the Gaussian) as an average of the points near the pixel where the area is half the total area of the Gaussian.

In 2.7 is easy to see how the two curves fits good enough to consider that our calculations are correct.
Finally, the program compares the obtained function of τ with different theoretical curves of τ obtained using 1.26 and 1.29 for different values of the chirp coefficient α. The equation in terms of α can be written as:

$$T_{ch} = T_0 \left(1 + \left(\frac{2\alpha x}{L_d} \right) + \left(\frac{x^2(1+\alpha^2)}{L_d^2} \right) \right)^{1/2} \tag{2.1}$$

With that comparison, one is able to observe not only if the measured tau curve is within the expected range, but to observe what is the possible chirp coefficient

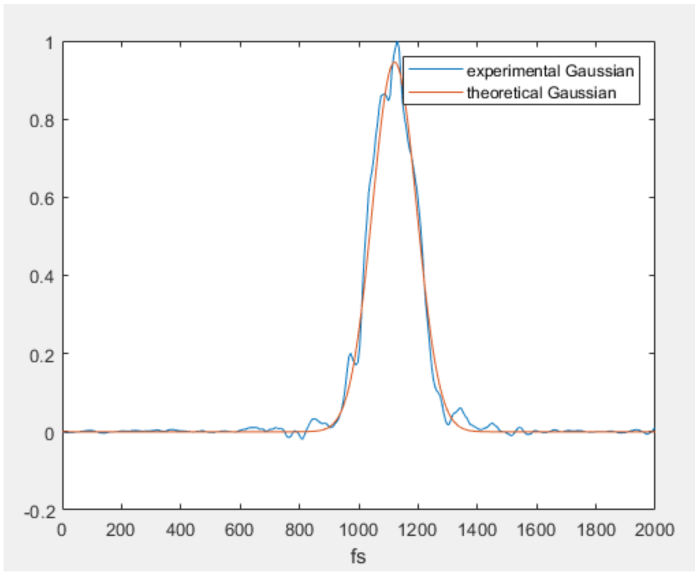


Figure 2.7: Comparative between the obtained in the laboratory and a theoretical Gaussian profile

value.

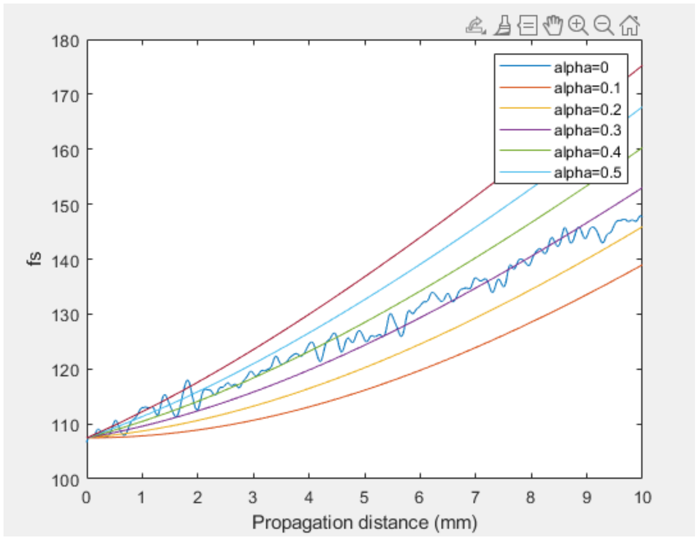


Figure 2.8: Comparative between the T obtained in the laboratory and theoretical Tth profiles for different α

In 2.8 one can observe that although the T function measured from the laboratory oscillates randomly along his graph, its behaviour fits quite well the different theoretical curves. Before obtaining this graph, we expected the measured T would be closer to the curve with $\alpha = 0$, so this results shows us that the laser is not perfectly calibrated or that there is some other distortion that affects the laser pulse. All in all, the Matlab program has proved to work successfully.

2.2 LabVIEW program

Labview is a system-design platform and development environment for a visual programming language from National Instruments, which is commonly used for data acquisition, instrument control, and industrial automation on a variety of operating systems, like Microsoft Windows. It was first released in 1986, and it has several versions, the program of this project has been built using the version of Labview 2018 SP1.

The most remarkable aspect of labview is that it uses graphical programming. It integrates the creation of user interfaces into the development cycle, what is one of the main aspects why it has been chosen for this project, because since the beginning of the programming process, the so-called front panel (the user interface) can be observed, tested and modified, and each VI (which are the LabVIEW programs-subroutines) can be easily tested before being embedded as a subroutine into a larger program.

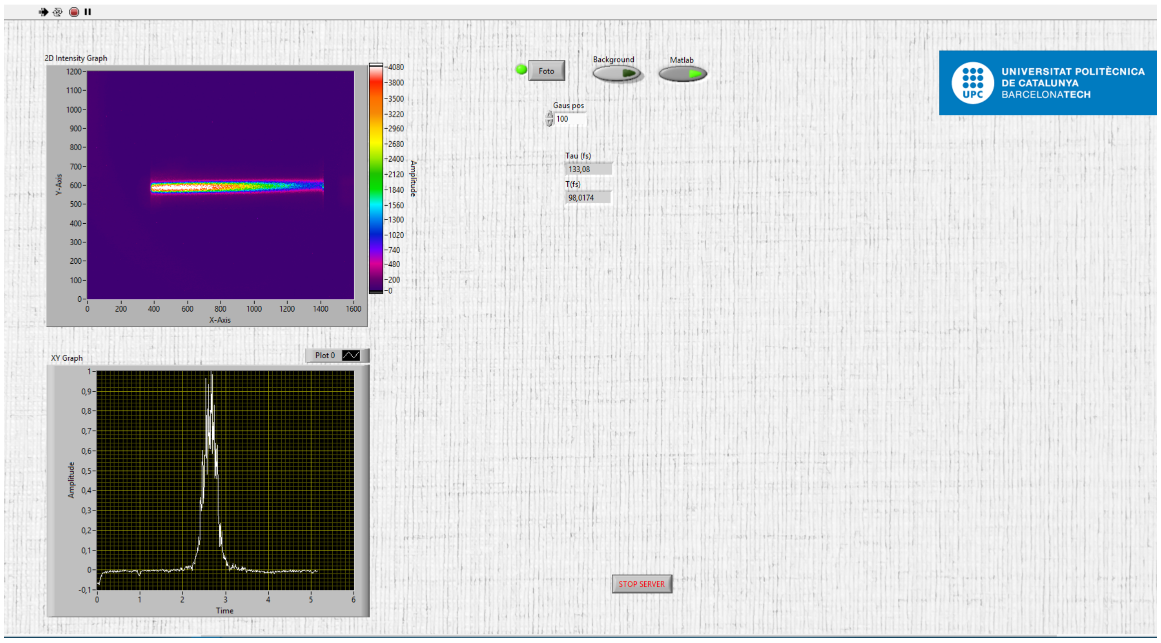
The other important benefit that made LabVIEW be the final choice for this project is its extensive support for interfacing to devices, like cameras. It has many libraries with numerous functions for data acquisition, mathematics etc. Furthermore, it is simple to integrate Matlab codes in LabVIEW, thus saving a lot of time to translate the Matlab program to other programming language.

Both Matlab and LabVIEW has the inconvenience that are not free software, but since UPC software distribute licence to their students, this does not suppose a problem.

Anyway, LabVIEW has inconveniences. It has very bad compatibility between different versions of the same program, specially when using external programs in the VI. If the version of an external software actualizes, the VI needs to be rebuilt and change all the libraries for the newer ones. This can be a huge problem and makes it necessary to work always in all the computers with the same versions of every software used in the program.

All in all, the pros of LabVIEW outweigh the cons, and this is the reason why LabVIEW was chosen instead of other free software. Anyway, as it was stated for the Matlab part, one of the main goals to continue this work is to translate this program to a free-license software, like Arduino, which will be tried in the near future.

Once the introduction on why LabVIEW was the chosen option is done, the program will be explained below in detail. Firstly, the script needs to capture in real time the image from the camera, to do so, LabVIEW reads the input from BeamGage. BeamGage is a software that shows in real time the image being captured by a camera connected to the computer, and allows applying many filters to the observed image. The program, as expected, allows saving captures of the shown image in different formats. Using different BeamGage LabVIEW libraries in different steps, LabVIEW is able to open BeamGage, inject an initial setup, and project what BeamGage is showing in the program. That means that one has to open the BeamGage on the computer after LabVIEW opens it. That could be a problem to be solved, as one could expect the same LabVIEW to do all the job, so the user does not need to use at the same time different software products. Anyway, this initialization part was taken from another project from years ago and only the work to translate it to another LabVIEW version (the initial program was built in LabVIEW 2013 and this project was realized in LabVIEW 2018) and also change the BeamGage libraries to the newest ones (because the BeamGage version was also actualized) was a heavy work. So after many problems and different versions, when the VI that initializes the BeamGage start working in the desired versions of both programs we realized that the program would be faster and easier for the user (who usually is already familiarized with the BeamGage software) if the filters and effects were previously selected directly from BeamGage. After selecting the desired camera and filters in BeamGage, all the rest of the program works entirely inside Labview. From here, the block diagram splits in two parts, both regulated by a Start/stop mechanism, which is the photo button. When the user clicks the photo button, the first half of the block diagram is in charge of capture the image itself and translate it to a real 2D matrix. The second half is more complex. It consists of a loop inside a switch case. The loop has two important parts, the first one does save the matrix obtained from the image in a .txt file. This part has a button that the user can press to distinguish when the photo is the background or the signal itself, thus letting the user change the background when it is necessary. The only change that exists between the two cases is that the saved file is named ”background.txt” or ”signal.txt” in order to let the Matlab code import both matrix separately. The next part from the loop is the Matlab code itself. Thanks to the Matlab script from LabVIEW a Matlab code can be directly integrated in the VI. Furthermore, some inputs and outputs can be inserted and taken out from the Matlab script. There's an input called "Gaus pos" that lets the user of the program select from which pixel from the original image wants to see the Gaussian profile and its theoretical comparative. And there are some outputs, like the value of the initial T and τ, and the Gaussian curve at the entrance of the crystal. This loop stops either when the program is forced to stop, or when the user presses the Stop Server button, that makes the program stop running and closes the BeamGage too. The final version of the program is shown in 2.9



Gaus pos 100

Tau (fs) 133,08

T(fs) 98,0174

Figure 2.9: Final version of the LabVIEW program

As it is shown in 2.9, at the top left of the program the image that is capturing the camera in real time is shown. Below the camera image, there is the Gaussian profile of the image in the pixel position selected by the control "Gaus pos". At the right of the image there is the "Foto" button, which takes a photo and starts the loop of the program, as expected. Next to it there are the Background and the Matlab buttons. The background button indicates when the next photo to be taken is the Background image. The Matlab indicates if, after taking an image, the program does the calculations and show the resulting graphs and results. These buttons do nothing by themselves, but they need to be selected before clicking the "Foto" button. It is important to observe that the Matlab button needs to be pressed after taking the initial background in order to pass the Matlab the two photos it needs (it is also possible to take first the signal image and then take the next photo with both the background and Matlab buttons selected). The Stop Server button ends the program, as explained before.

Chapter 3

Results

Finally, with the program finished and fully functional, the last step consists in measure 3 different pulses, analyse the obtained results and check if they are correct. It is needed to say that at the beginning of this project the idea was to use this program both in the femtosecond and picosecond regime lasers, but unfortunately, the picosecond laser broke so all the measures are done with the same femtosecond laser. Anyway, the 3 pulses will be different from each other, since the initial pulse is modified using different techniques in order to obtain measures of pulses with different behaviours.

3.1 140fs pulse

This pulse is the common pulse emitted by this laser, it is emitted and sent directly to the measurement set-up described in 1.6.2. The pulse was measured with the program and the obtained results were the following:
The measured $\Delta z(0) = 0.3519mm$, and with that, the obtained duration of the pulse was 108.4371fs, thus meaning that the obtained τ is 127.6749fs. From 3.1 one can observe that the chirp coefficient is between 0.1 and 0.3. The obtained values are correct enough. Although the laser is theoretically emitting at 140fs, this can not be exact, as this ultrafast lasers require very precise conditions to work properly, and they can be bad aligned, so we can conclude that this result is valid.

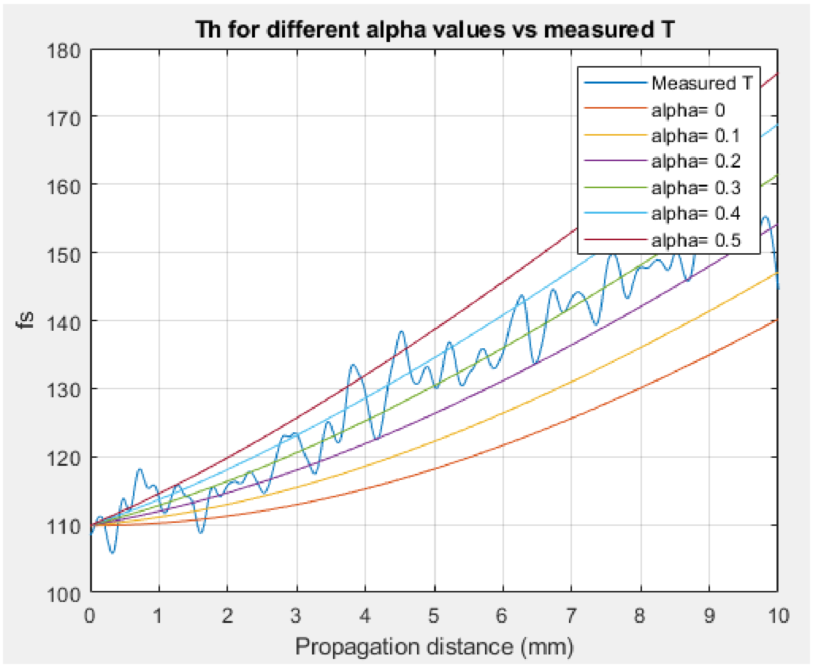


Figure 3.1: Obtained T with different T_{th} for different chirp coefficient values

3.2 Bad calibrated pulse

Now we aligned the used laser badly on purpose in order to observe how the program observes this bad behaviour and that the results are not the same as the previous experiment. Before using the program and observe the results, we expect this pulse tu be wider at the beginning of the crystal.

The measured $\Delta z(0) = 0.3462mm$ which leads to a pulse duration of $T = 142.2516fs$ and a $\tau = 167.4885fs$. Here it is so hard to say which is the chirp coefficient, since the fact that the laser is bad aligned makes that it starts to decrease its duration before going out of the crystal, as can be observed in 3.2. This is due to the fact that here the chirp coefficient is increased at the entrance of the crystal, thus making that the pulse gets wider faster and the overlap between the two incoming beams is not maintained during all the crystal length. The part of the graph starting when the beam width starts decreasing is not representative. Obviously, this result is different from the previous one. Furthermore, the pulse is wider than the original one, as we expected before making the experiment, and the fact that it decreases its temporal duration from a certain point of the crystal fits also with the expected results.

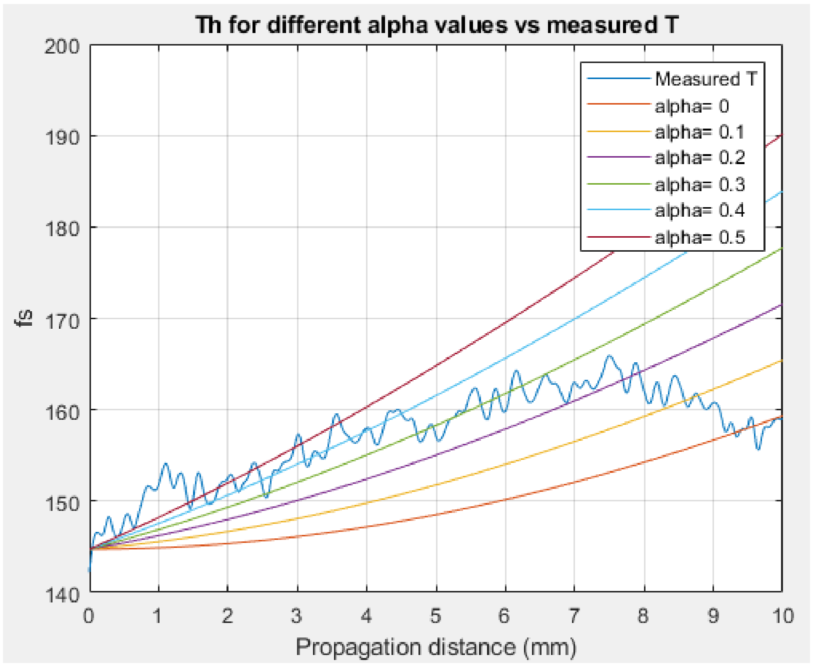
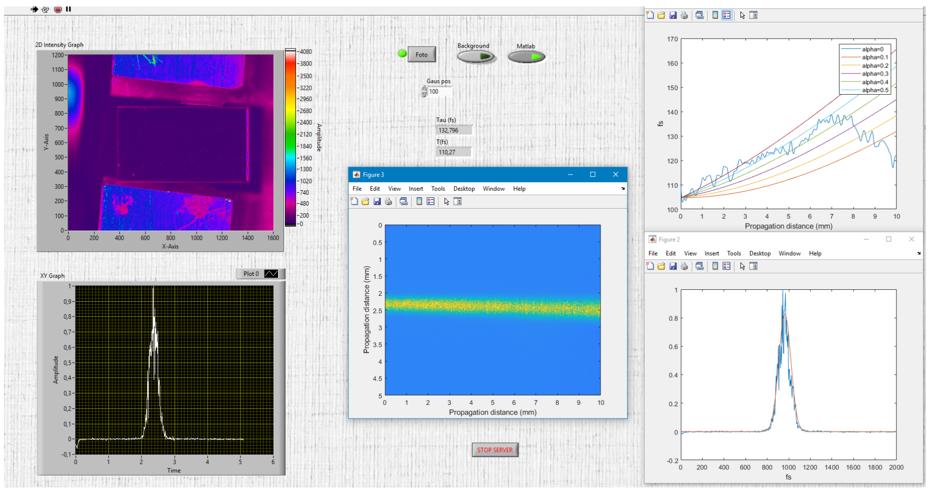


Figure 3.2: Obtained T with different T_{th} for the laser bad aligned



Gaus pos 100
Tau (fs) 132,796
T(fs) 110,27

Figure 3.3: Obtained results from the program for the bad signal

3.3 3 peak pulse

In order to test further the set-up, we designed and mounted a setup to obtain a more complex pulse consisting of a two-pulse train with a selected separation between them. To achieve this goal, we constructed a Michelson interferometer to generate two replicas from the incident pulse. By controlling the delay of one interferometer arm, we can change the separation between the replicas, obtaining a two-peak pulse which was sent to the autocorrelator for its measurement. The profile of that interaction shows one central Gaussian profile with two little Gaussian at each side of the central peak, as is shown in 3.4.

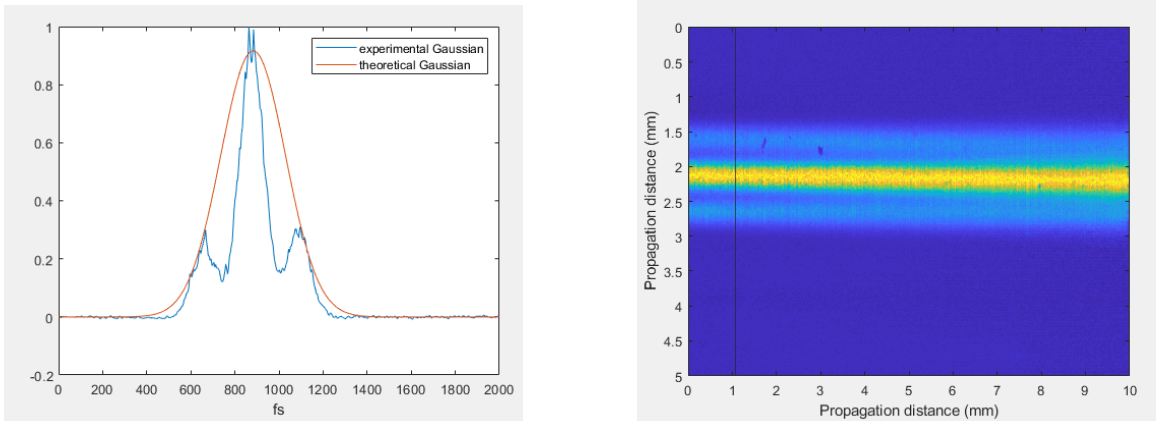
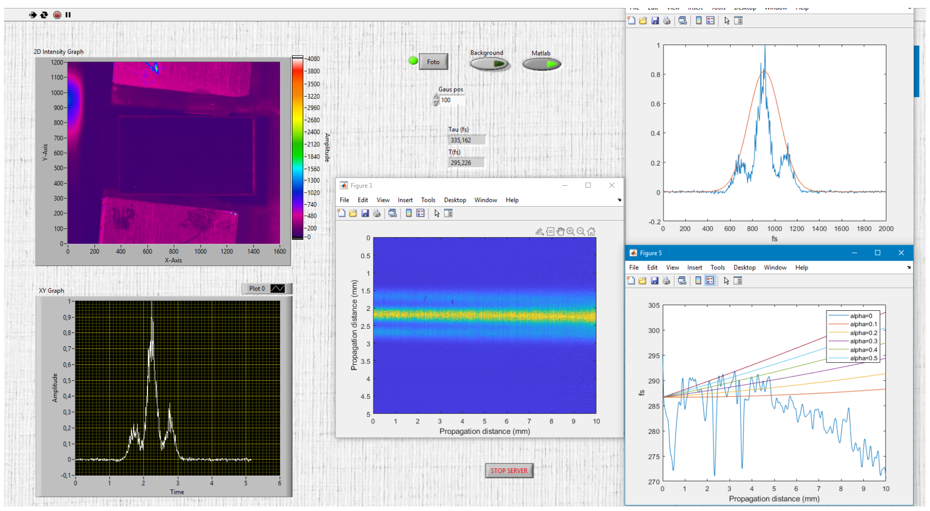


Figure 3.4: Image and profile of two pulses forming a 3 peak profile

This pulse should not be measured correctly, as the program assumes that the lasers are Gaussian-shaped and the resulting pulse of this interaction is definitely non-Gaussian. In fact, we are working with 2 pulses here, and the program is not ready to expect more than a pulse at the same time. The obtained results are the following.
The measured $\Delta z(0) = 0.6721mm$ which leads to a pulse duration of $T = 276.1355fs$ and a $\tau = 325.1247fs$. The program here reads the incoming signal as a unique pulse, but as it is in fact a 3 peak pulse due to the interaction of the 2 original pulses, the pulse duration is higher than the original pulse.



Gaus pos
100

Tau (fs)
335,162

T(fs)
295,226

Figure 3.5: Obtained results from the program for 3 peak signal

Again we can say that the program responds correctly, as it is not programmed to read 2 pulses at the same time, but it does a logical interpretation of this incident signal.

Chapter 4

Discussion and Future Perspectives

The results obtained and presented in this work show that it is possible to build a program that allows to automatize the pulse duration measurement process for femtosecond lasers. Despite the program present some limitations, the femtosecond pulses emitted by the laser can be correctly measured and analysed. The program is able to show in real time the image captured by the camera, plot the evolution of the desired parameters, like the pulse duration and the chirp coefficient, along the crystal, show the Gaussian profile at a selected by the user point of the image, etc. Other parameters or graphs can be obtained, but these were the selected following the requirements of the lab team. The program is easy to read and modify, what means that it can be used by different people for different lasers in the future.

By looking at 2.4 and 2.6 one may notice that different calibration processes need to be done before showing the results if one wants to obtain a proper solution that does not contain too much distortion or error on it. Thus, as the computational cost is limited since the program needs to be fast in order to be useful in the laboratory, it is extremely important for the algorithm to be optimized as possible. Besides these limitations, the final program is fast enough and is able to obtain the results with the desired precision and without much distortion, as can be seen in 2.7.
As shown in 2.1, the calibration process lead also to some debate. The original idea was to use the autocorrelation process to change the units of the system to measurable units instead of pixels, but we had found with the same problem as when trying to select the important part of the image. Non image recognition was implemented in this work as it did not seem feasible, thus meaning that there is no way left to count the pixels between the different lines from 2.6 that by looking

directly at the image. Anyhow, this was not a problem since using the length of a crystal as a reference leads to the same results.

Another of the biggest limitations in this program is the LabVIEW itself and its bad compatibility with external program's libraries. One of the toughest works in this project was to open the BeamGage inside the LabVIEW program in the correct version using the correct libraries and been able to take a photo in the correct format. Furthermore, although first one may think that it is so easy to implement Matlab code inside the LabVIEW script, this problem is harder than it seems. The reason is not clear yet, but sometimes LabVIEW is not able to send an input value or take an output value from the Matlab script. This problem seems to be common for many LabVIEW users, but after many versions the final program does not present this problem any more.

In the future, one of the first objectives is translating this program to a free-source software. Besides that, this program could be tested in other laser regimes, specially in the picoseconds' regime, where it should work also perfectly. Implementing the image processing would be also a great add, since it would allow working with different set-ups and moving the camera without need of changing manually the image selection part of the Matlab code.

Chapter 5

Conclusion

The thesis has turned in a successful implementation of the theoretical photonics measurements methods in a user-friendly program. A program that is able to calculate some important parameters of the pulse, specially the pulse duration has being created and modified in such a way that it results easy for the user to work with and interpret the results. This program has been designed for a particular set-up from the laboratory that implements 1.6.2. The set-up was also correctly understood theoretically and some modifications and manual measurements have been also realized in order to understand how much effort working in an experimental physics laboratory means.

After the initial theoretical study of non-linear optics and the different Second harmonic generation measurement methods for ultrashort laser pulses, the problems that the desired program needed to solve were presented. After importing the image in the correct format and do the so-needed image processing to the obtained matrices, a numerical knife-edge method have been implemented in order to calculate the D_{16-84} profile of the incident pulse. After that this profile is translated to mm and fs, and the different desired profiles and values are obtained using the corresponding mathematical formulas described in 1.
This Matlab code is integrated in a LabVIEW script that allows the user to capture the image from the camera and obtain the results easily all in the same program.

Although, as written in the discussion and further perspectives section, further tests and experiments can be realized with the program, it seems safe to say that the main goal of this thesis has been achieved.

Bibliography

[1] J Trull, S Saltiel, C Cojocaru, et al. Characterization of femtosecond pulses via transverse second-harmonic generation in random nonlinear media. *Springer-Verlag*, 2009.

[2] Trebino R. Frequency-resolved optical gating: The measurement of ultrashort laser pulses[m]. *Springer Science Business Media*, 2000.

[3] Diels J C and Rudolph W. Ultrashort laser pulse phenomena[m]. *Academic press*, 2006.

[4] Bingxia Wang. Second harmonic generation in disordered nonlinear crystals: Application to ultrashort laser pulse characterization. *PhD thesis, Universitat Politècnica de Catalunya, Spain*, 2017.

[5] Fejer M. M., Magel G. A., Jundt D. H., et al. Quasi-phase-matched second harmonic generation: tuning and tolerances[j]. *IEEE Journal of Quantum Electronics*, 28(11):2631–2654, 1992.

[6] Roppo V, Dumay D, Jose Trull, Crina Cojocaru, et al. Planar second-harmonic generation with noncollinear pumps in disordered media[j]. *Optics express*, 16(18), 2008.

[7] R. Fischer, D.N. Neshev, S.M. Saltiel, W. Krolikowski, and Y.S. Kivshar. Broadband femtosecond frequency doubling in random media. *Appl. Phys. Lett.*, 2006.

Appendices

Appendix A

Appendix 0

```
 clc
clear all
%
b= 'C:\Users\Usuario\Documents\TFG\ac2pulsosbackground.txt';
[Back]=importdata(b);
g= 'C:\Users\Usuario\Documents\TFG\ac2pulsossignal.txt';
[Gaus]=importdata(g);
Gaus=Gaus-Back; %resto el background
Gaus=Gaus(330:820,375:1310); %Obtengo la parte de
% inter   s de la imagen40

for i=1:3
    Gaus=imgaussfilt(Gaus);
end
D1684=zeros(size(Gaus,2),1);
T=zeros(size(Gaus,2),1);
xo=zeros(size(Gaus,2),1);
for i=1:size(Gaus,2)
AT=trapz(Gaus(:,i)); %Area total
b1=1;
A=0;
while(A<=0.16*AT) %Calculo posicion Area=0.16*Area Total
    b1=b1+1;
    A=trapz(Gaus(1:b1,i));
end
b2=b1;
while(A<=0.84*AT) %Calculo posicion Area=0.84*Area Total
    if(A>=0.48*AT && A<=0.52*AT)
    xo(i)=b2;
    end
    b2=b2+1;
    A=trapz(Gaus(1:b2,i));
end
D1684(i)=b2-b1;
Gaus(:,i)=Gaus(:,i)/max(Gaus(:,i));
end



%%Paso D1684 a mm

xx=linspace(0,10,size(Gaus,2));
D1684=D1684*(5/size(Gaus,1));

for ii=1:10
D1684=smooth(D1684);
end

%Calculo el per  odo del laser
aext=5;
c=3e8;
T=sqrt(2)*sind(aext)*D1684/(1000*c);
T=(10^15)*T;
%
% figure(1)
% plot(xx,D1684);
% title('wo(z)')
% hold on

pos=100;


Io= (Gaus(xo(pos)-2,pos) + Gaus(xo(pos)-1,pos) + Gaus(xo(pos),pos));
Io=Io+ (Gaus(xo(pos)+1,pos) + Gaus(xo(pos)+2,pos))/5;
xx=linspace(0,5,size(Gaus,1));
xo=xo*(5/size(Gaus,1));
thgaus =Io*exp(-2*(xx-xo(pos)).^2./((D1684(pos,1))^2));
%Calculo una gausiana te  rica



%Comparo la gausiana te  rica con la obtenida de la imagen
figure(2)
title('Theoretical vs Obtained measure')
plot(10^(15)*xx*sqrt(2)*sind(aext)/(1000*c),Gaus(:,pos));
hold on
```

```
plot(10^(15)*xx*sqrt(2)*sind(aext)/(1000*c),thgaus);
xlabel('fs')
xlim([0,2000])

legend('experimental Gaussian','theoretical Gaussian')


figure(3)
imagesc(linspace(0,10,size(Gaus,2)),xx,Gaus)
hold on
xline(pos*10/size(Gaus,2));
xlabel('Propagation distance (mm)')
ylabel('Propagation distance (mm)')


figure(4)
xx=linspace(0,10,size(Gaus,2));
plot(xx,T);
title('T(z)')
xlabel('Propagation distance (mm)')
ylabel('fs')
ylim([100,200])


%%Calculo les corves de tau en funci  de alpha
 tau=T*sqrt(2*log(2));
g=466;%fs/mm
Ld=(T(1)^2)/(2*g);
To=0;
for j=1:10
    To=To+T(j);
end
To=To/10;
figure(5)
plot(xx,T,'DisplayName',['Measured T'])
xlabel('Propagation distance (mm)')
ylabel('fs')
title('Th for different alpha values vs measured T')
hold on
grid on
for alpha=0:0.1:0.5
Tth=To*sqrt(1+(2*alpha*xx/Ld)+(xx.^2.*(1+alpha^2)/(Ld^2)));
plot(xx,Tth,'DisplayName',['alpha= ',num2str(alpha)])
end
legend show
```

```
beta=466;

T(1)
tau(1)
```