Bachelor Thesis
Short-range Wakefield Model Implementation in OPAL

Stefan Pauli

June 26, 2008

Supervisors: Dr. A. Adelmann
Prof. Dr. P. Arbenz

ETH Zürich
Inst. of Computational Science

Abstract

The Paul Scherrer Institut (PSI) is planning a new X-ray free electron laser (XFEL). Complex simulations must be done before constructing this machine. The Object Oriented Particle Accelerator Library (OPAL) is used for accurate 3D simulations. To that end, new features have to be introduced into OPAL. One of them are short range wakefields, the topic of this bachelor thesis. First a stand alone prototype was built, documented and tested against an already existing Mathematica notebook. After having passed the tests the prototype was incorporated into OPAL.

Contents

1 Introduction

The goal of this bachelor thesis was to add short range geometric wakefields to the Object Oriented Parallel Accelerator Library (OPAL).

1.1 Short Range Wakefields

In special cases of physics of linear accelerators and storage rings one can consider particles at almost speed of light. This leads to the concept of wakefields and wake potentials. In this concept particles are considered as point charges moving at a velocity close to the velocity of light. If such a point charge would move in the free space, its electric and magnetic field would lie nearly in a plane passing through the charge and perpendicular to its path. This can be shown in a laboratory frame. This means that a second particle which follows the first one with the same velocity on a parallel path will not be subjected to any forces from the fields produced by the first one. But in a linear accelerator or in a storage ring the particles are surrounded by a beam pipe, which changes the boundary conditions of those two charges. In this configuration the first particle will not be subjected to any forces from the fields produced from the second one, because of causality. But the trailing charge feels indirectly the field of the wavefront moving with the leading charge. Because the wavefront from the first particle will be scattered at the boundary. This scattered radiation will reach the second charge and exert a force parallel and perpendicular to its direction of motion. This scattered radiation are called wakefields. The integrated effect of these wakefields over a given path length of the trailing charge is called longitudinal and transversal wake potential. One way of computing the wakefield and the forces acting on the charges due to the wakefield has been implemented in OPAL in the framework of this the bachelor thesis.

1.2 OPAL

OPAL has been developed at PSI and is used to track particles in a variety of accelerators, including cyclotrons and linear accelerators. The official homepage of OPAL (`http://amas.web.psi.ch`) describes OPAL as follows: "OPAL (Object Oriented Particle Accelerator Library) is a C++ framework for general particle accelerator simulations. It includes various beam line element descriptions and methods for single particle optics, namely maps up to arbitrary order, symplectic integration schemes and lastly time integration. OPAL is based on IPPL (Independent Parallel Particle Layer) which adds parallel capabilities. Main functions inherited from IPPL are: structured rectangular grids, fields and parallel FFT and particles with the respective interpolation operators. Other features are, expression templates and massive parallelism (up to 8000 processors) which makes is possible to tackle the largest problems in the field."

1.3 Importance of Wakefields in PSI Simulations

The main reason to include wakefields into OPAL is the necessity to simulate them for the new X-ray free electron laser (XFEL). This is a linear accelerator. Ther operates with very high peak currents i.e. the particle bunch is very dense. The denser the particle bunch is, the bigger is the force due to the wakefield. Therefore, to get reliable results when simulating the new XFEL wakefields nead to be included.

2 Wakefield Calculation

2.1 Analytical Computation of Wakefields

By inverse FFT of the beam pipe impedance the wake function is determined. For special geometries of the beam pipe analytical results of its impedance are known. The round, metallic beam pipe with radius a is one example of a known impedance. There exist models of beam pipes with dc conductivity and models for ac conductivity. The dc conductivity of a metal is given by $\sigma = ne^2\tau/m$ with n the density of conduction electrons, e the electron charge, τ the relaxation time, and m the electron mass. The ac conductivity, a response to applied oscillation fields, is given by $\tilde{\sigma} = \frac{\sigma}{1-i\omega\tau}$ with ω the frequency of the fields. According to [2] the longitudinal impedance with dc conductivity is given by

$$Z_{Ldc}(k) = \frac{1}{ca}\frac{2}{\frac{\lambda}{k} - \frac{ika}{2}} \tag{1}$$

where

$$\lambda = \sqrt{\frac{2\pi\sigma|k|}{c}}(i + \mathrm{sign}(k)) \tag{2}$$

with c the speed of light and k the wave number. From [1] a Mathematica notebook for calculating the impedance is given. Inserting $Z_0 = \frac{4\pi}{c}$ and a (in units of mm) in equation (1) leads to the form of the dc impedance in [1]:

$$Z_{Ldc}(k) = \frac{Z_0}{2\pi 10^{-3}a}\left(\frac{\sqrt{\frac{\sigma Z_0|k|)}{2}}(i + \mathrm{sign}(k))}{k} - \frac{ik10^{-3}a}{2}\right)^{-1}. \tag{3}$$

This has the advantage that numerical results are available to compare the own results. The ac Impedance is given by [1]

$$Z_{Lac}(k) = \frac{Z_0}{2\pi 10^{-3}a}\left(\frac{\sqrt{\frac{Z_0|k|)}{2}}\sqrt{\frac{\sigma}{1-ick\tau}}(i + \mathrm{sign}(k))}{k} - \frac{ik10^{-3}a}{2}\right)^{-1}. \tag{4}$$

One can obtain the longitudinal wake by an inverse Fourier transformation of the impedance. Since $Re(Z_L(k))$ drops at high frequencies faster than $Im(Z_L(k))$ the cosine transformation can be used to calculate the wake. This equation holds for the dc and the ac case.

$$W_L(s) = 10^{-12}\frac{2c}{\pi}Re\left(\int_0^\infty Re(Z_L(k))\cos(ks)dk\right) \tag{5}$$

with $Z_L(k)$ either $Z_{Ldc}(k)$ or $Z_{Lac}(k)$ dependent whether the dc or the ac case is used.

The Panofsky-Wenzel theorem states

$$Z_L(k) = \frac{k}{c}Z_T(k). \tag{6}$$

Inserting the Panofsky-Wenzel theorem in equation (5) the transversal wakefield reads

$$W_T(s) = 10^{-12}\frac{2c}{\pi}Re\left(\int_0^\infty Re(\frac{c}{k}Z_L(k))\cos(ks)dk\right). \tag{7}$$

2.2 Numerical Computation of Wakefields

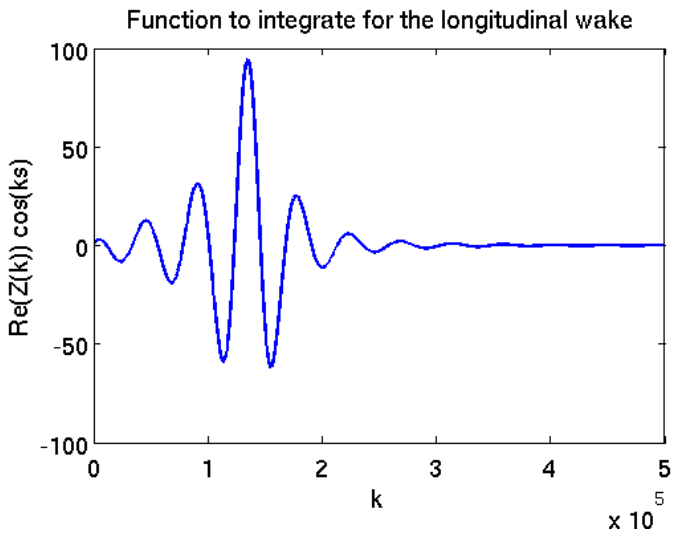

Figure 1: The important part of the integration lies between k=0 and k=300000

As shown in equation (5) and (7) one needs to integrate a function to calculate one sampling point of the wakefunction. For the numerical integration equation (5) is written as a sum from 0 to $N-1$ since ∞ can not be covered numerically, and a

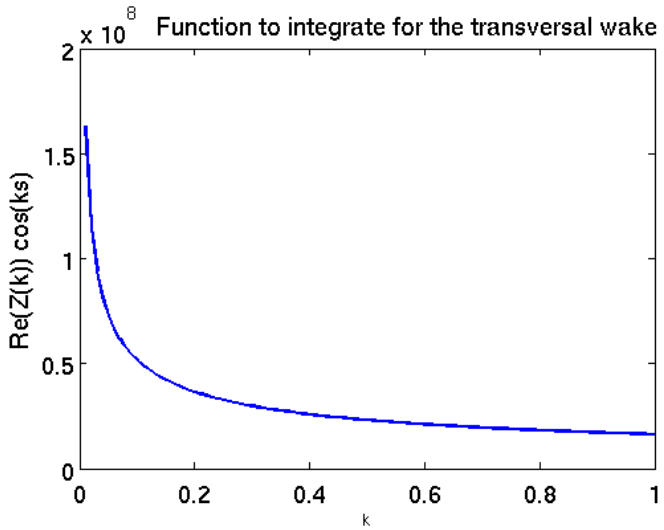

Figure 2: A small Δk is required to integrate this function

discrete Δk instead of a infinitesimal dk:

$$W_L(s) = 10^{-12}\frac{2c}{\pi}Re\left(\sum_{i=0}^{N-1} Re(Z_L(i\Delta k))\cos(i\Delta ks)dk\right) \tag{8}$$

with $Re(Z_L)$ either Z_{Ldc} or Z_{Lac}. The same holds for equation (7) which is then writen as

$$W_T(s) = 10^{-12}\frac{2c}{\pi}Re\left(\sum_{i=0}^{N-1} Re(\frac{c}{k}Z_L(i\Delta k))\cos(i\Delta ks)dk\right). \tag{9}$$

To have reliable results out of equation (8) and (9) N the number of sampling points must be big enough and Δk the mesh size small enough. Figure (1) and Figure (2) give an indication how to choose N and Δk. in both figures the parameters are chosen for a round, copper beam pipe with radius 5 mm. In figure (1) $Re(Z_{Ldc}(k))\cos(ks)$ with s=120 μm is plotted with respect to k. This equation is integrated in equation (8) to get the longitudinal wakefield. There it is obvious that k must be at least in the order of 10^5. In figure (2) $Re(\frac{c}{k}Z_{Lac}(k))\cos(ks)$ with s=120 μm is plotted with respect to k. This equation is integrated according to equation (9) to get the transversal wakefield. Figure shows a singularity at k=0. To integrate such functions a small Δk (fine mesh) is needed. To simplify the understanding and the implementation of the code it would be nice to have the same integration schema for both wakes. In this bachelor thesis the Simpson integration schema with equidistant mesh size was used. This leads to an integration with small Δk with a big N which is computational not so efficient. Since the calculation of

the wakefield is usually just made once in the initialization phase of the program this does not hurt the performance of the program so much.
It would make sense to change to a schema with adaptive mesh size to improve the efficiency of the wake calculation.

2.3 Using other Wakefields

The application area for wakefield generated by round, metallic beam pipes with a certain radius is limited. Therefore one needs to be able to integrate in the code different wakefields leading to the possibility to simulate the behavior of particles in any kind of beam pipe. This is solved by the possibility to load the data from any $W_L(s)$ and $W_T(s)$ in to OPAL. According to section 2.4.1 values of $W_L(s)$ or $W_T(s)$ are just needed at several discreet points for s to calculate the force on the particle. If the data loaded in to opal does not provide the function values at those points, the function values are interpolated in a linear way. This makes it possible to calculate in external programs any kind of wakefields and load them in to OPAL. Hence simulations can be done with any kind of beam pipes.

2.4 Calculate the Force on the Particles

If one has the line density λ of the particle bunch and the wakefield one can compute a force acting on the particle. The wakeforce parallel to the direction of motion of the particle can be calculated using equation (10), the force perpendicular to its direction of motion can be computed using equation (11) and (12) given by [3].

$$F_x(s) = q \int_0^\infty W_T(s - s')x(s')\lambda(s')ds' \tag{10}$$

$$F_y(s) = q \int_0^\infty W_T(s - s')y(s')\lambda(s')ds' \tag{11}$$

$$F_z(s) = q \int_0^\infty W_L(s - s')\lambda(s')ds' \tag{12}$$

with z parallel to the direction of motion of the particle, x(s) the shortest distance between the center of the beam pipe and the particle in x direction and y(s) the shortest distance between the center of the beam pipe and the particle in y direction.

2.4.1 Discrete Calculation of the Force

In the simulation space an time are discrete. Hence space is divided in finite slice. Particles being in the same slice feel the same force. Therefore it is reasonable to calculate the force exert for every slice. This calculation is called energy spread in [1].

Space discrete signals are used to write equation (10) to (12) in a discrete form. Signals in the from $x[n]$ are defined as $x[n] = x(s)\mid_{s=n\Delta s}$.

$$F_x[n] = q\sum_{i=0}^{N-1} W_T[n-i]x[i]\lambda[i] \tag{13}$$

$$F_y[n] = q\sum_{i=0}^{N-1} W_T[n-i]y[i]\lambda[i] \tag{14}$$

$$F_z[n] = q\sum_{i=0}^{N-1} W_L[n-i]\lambda[i]. \tag{15}$$

A convolution can be calculated with FFT and IFFT with $\mathcal{O}(n\log n)$ instead of $\mathcal{O}(n^2)$. Therefore it is a goal to bring those equations in the form of a convolution $y[n] = \sum_{i=-N}^{N} a[n-i]b[i]$. This can be achieved with th Heaviside step function

$$H[n] = \begin{cases} 0, & n < 0 \\ 1, & n \geq 0 \end{cases} \tag{16}$$

where the equations can be written as

$$F_x[n] = q\sum_{i=-N+1}^{N-1} W_T[n-i]H[n-i]x[i]\lambda[i]H[i] \tag{17}$$

$$F_y[n] = q\sum_{i=N+1}^{N-1} W_T[n-i]y[i]\lambda[i]H[i] \tag{18}$$

$$F_z[n] = q\sum_{i=N+1}^{N-1} W_L[n-i]\lambda[i]H[i]. \tag{19}$$

Using FFT this lead to the form implemented in the program. The transformation pair is written as

$$\vec{F_x} = q\mathrm{IFFT}(\mathrm{FFT}(\vec{W_T})\mathrm{FFT}(\vec{x}\vec{\lambda}\vec{H}) \tag{20}$$

$$\vec{F_y} = q\mathrm{IFFT}(\mathrm{FFT}(\vec{W_T})\mathrm{FFT}(\vec{y}\vec{\lambda}\vec{H}) \tag{21}$$

$$\vec{F_z} = q\mathrm{IFFT}(\mathrm{FFT}(\vec{W_T})\mathrm{FFT}(\vec{\lambda}\vec{H}) \tag{22}$$

where for all signals holds $\vec{x} = \{x[-N+1], x[-N], \cdots, x[N-1]\}$.

3 Build a Prototype

Our prototype is a stand alone application which calculates wakefields and computes the resulting forces on the particle.

3.1 Why a Prototype

According to the OPAL doxygen documentation OPAL consist of 842 program files and has 492 classes. This gives a rough idea how complex and big OPAL is. To get to know such a huge program takes a while. To learn in the same time things about wakefield and implementing them would be to much. Therefore it made sense to partition those topics in different steps. One good way is to first make a stand alone application which implements most of the tasks of this bachelor thesis. Towards the end of the bachelor thesis this prototype can be ported to opal. In this way one can first understand, implement and test the wakefunction without the complexity of opal. Afterwards by the implementation in OPAL the complexity is reduced since one has a already tested program to port to OPAL.

3.2 Approach

First a Prototype was implemented in MATLAB. This has the advantage that it is quite fast coded and that one can make plots of everything very easy. After the MATLAB Prototype was validated (see section 4) a prototype in C++ was made. Since OPAL is also written in C++ this prototype will be well portable to OPAL.

3.3 Functionality of the Prototype

3.3.1 Calculate the wakefield

Calculating the wakefield of a round, metallic beam pipe was the first task of the prototype. In section 2.1 the needed functions are given for bough the transversal and the longitudinal wakefield. In the prototype one can choose between a copper and a aluminum beam pipe with arbitrary radius. This is enough for testing the prototype, but one could easily expansion this to other materials.

3.3.2 Calculate the Force

The next implemented functionality is the calculation of the force acting on the particles. The equations are given in section 2.4. In this equations a particle line density λ is used. In the prototype one need to assume a particle density. Since the space is discretized particles being in the same slice feel the same force. Therefore it is reasonable to calculate the force exert for every slice. This calculation is called energy spread in [1].

This calculation of the force will be made in opal in every step. Therefore it is very important to make the computation of the force as chap as possible. As discussed in section 2.4.1 the force can be calculated using FFT. This takes only $\mathcal{O}(n\log(n))$ instead of $\mathcal{O}(n^2)$ computation effort.

3.3.3 Two Particle Model

The Leap Frog algorithm is used to implement the two particle model. One of those two particle is the whole particle bunch and the other particle is a single particle. The bunch is just negligible influenced by the single particle. For this reason just the force on the single particle is calculated. Therefore the form of the bunch does not change, and all the particles from the bunch have a constant velocity of β. The velocity of the single particle is time variant. Therefore an integration method is used to calculate position and velocity of the single particle after a certain time t. To conserve energy the Leap Frog algorithm is chosen.
This calculation of the new position of a particle can be done seperatly for all start positions of the particle. Then a plot is provided with all this particle position relative to the particle position without wakefields. This functionality is only integrated in the MATLAB code.

3.4 Documentation

A proper documentation is required if a code should be useful for other persons. To make live easy for the user the documentation should use standard formats. In MATLAB it is common that you can access to the documentation with typing help filename in the command window. In C++ doxygen documentation is common and also used in opal. Therefore those documentation forms were chosen. The documentation of the code is only made there. Hence there is no code documentation in this document. To give a short idea how the documentation looks like here two examples, one for MATLAB and one for the C++ code. Figure 3 shows the result

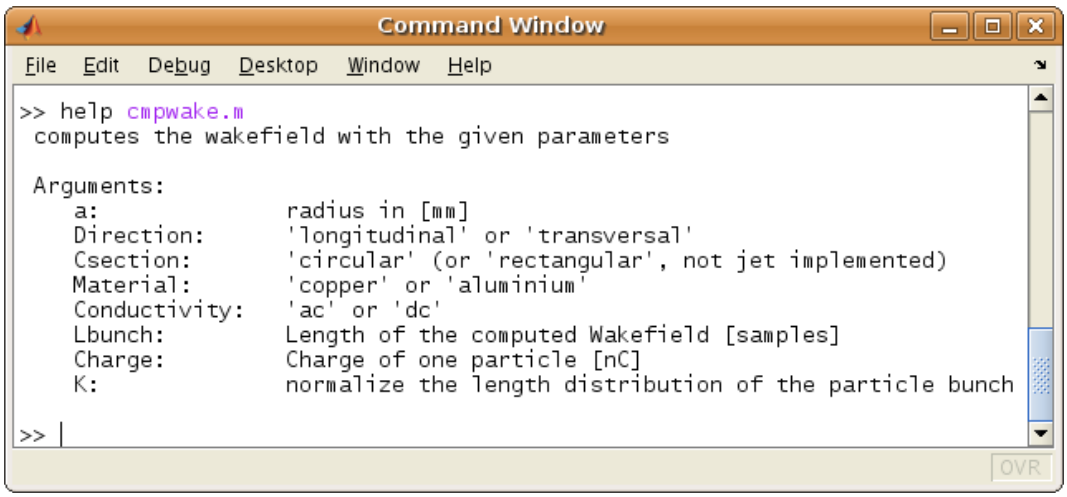

Figure 3: Example of the MATLAB prototype documentation

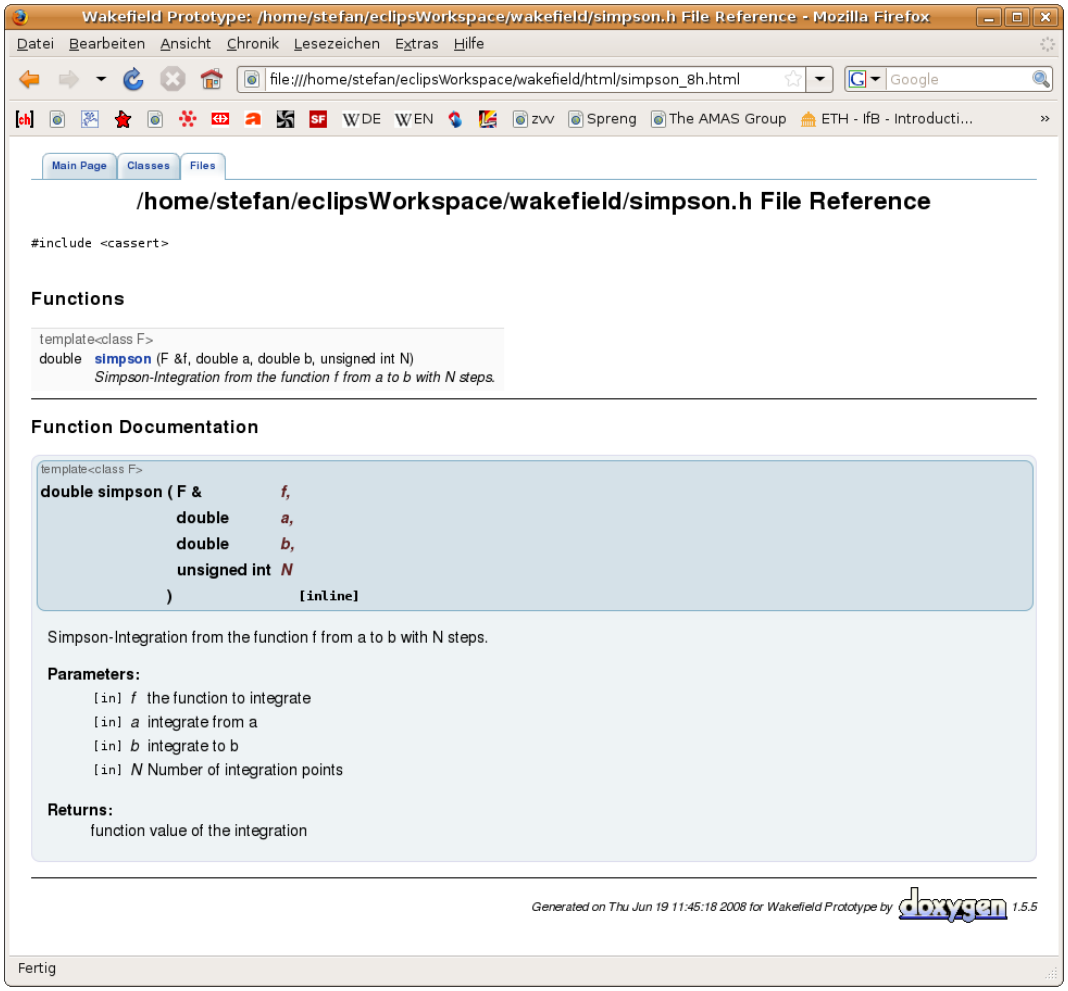

Figure 4: Example of the C++ prototype documentation

when typing help cmpwake.m in the MATLAB command window. Figure 4 shows a example of the C++ code documentation.

4 Validation of the Prototype

4.1 Comparison with the Mathematica code

One way of validating the prototype is to compare its results with the results of an existing Mathematica code [1]. This Mathematica code computes a wakefield and a energy spread. For comparison the energy spread an the wakefield are compared with different settings.

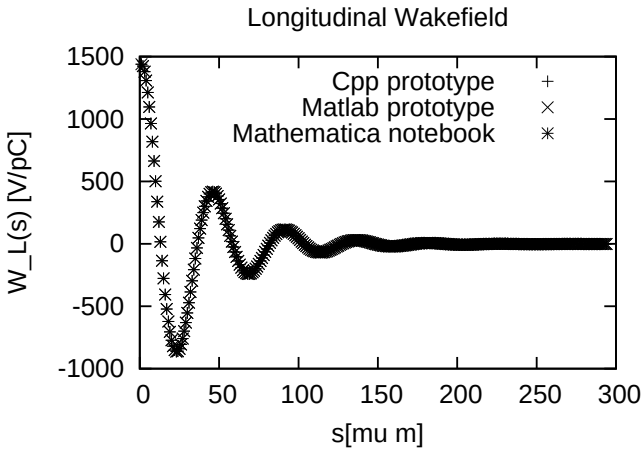

Figure 5: Comparison of the MATLAB and the C++ prototype with the Mathematica code from [1]. This is the longitudinal wakefunktion of a round, copper beam pipe with radius 5 mm and ac conductivity.

Figure 5 shows that the result is equal for the prototype and the Mathematica code for the longitudinal wakefield for a round, copper beam pipe with radius 5 mm and ac conductivity. This holds for all wave function tested with the 3 codes. Figure 6 is a second example where the prototype agrees with the Mathematica code.

A comparison between the energy spreads of the Mathematica code and the prototype in C++ or MATLAB with always the same result can bee seen in Figure 7.

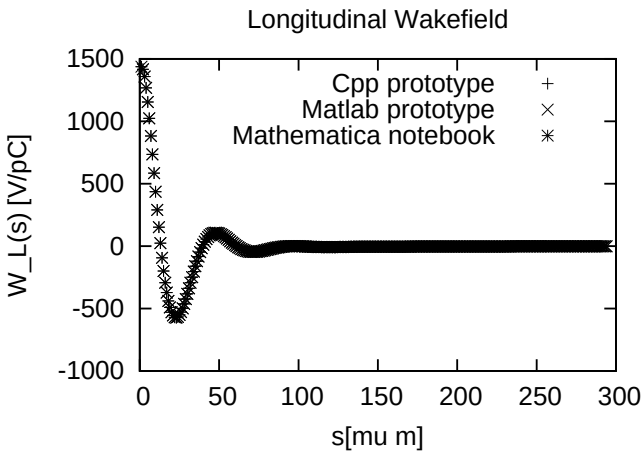

Figure 6: Comparison of the MATLAB and the C++ prototype with the Mathematica code from [1]. This is the longitudinal wakefunktion of a round, aluminum beam pipe with radius 5 mm and ac conductivity.

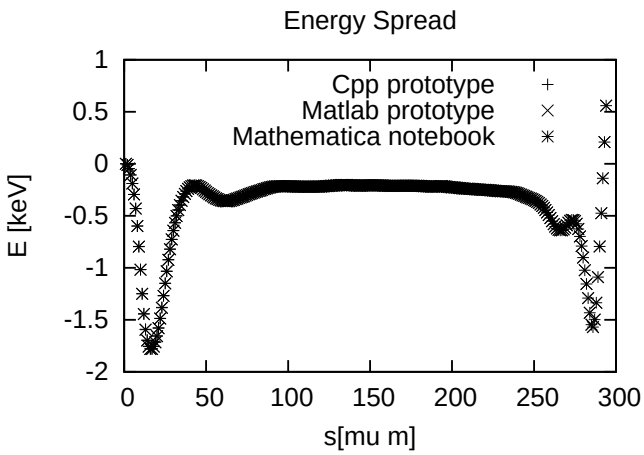

Figure 7: Comparison of the MATLAB and the C++ prototype with the Mathematica code from [1]. This is the so called energy spread in [1] resulting from a longitudinal wakefunktion of a round, aluminum beam pipe with radius 5 mm and ac conductivity with the line density λ from [1].

5 Implementation in OPAL

The C++ prototype was ported to OPAL. Hence most of the functionality is the same as in the prototype. Topics already discussed in connection with the prototype are omitted in this section to avoid repetition. New topics valid only for the implementation of OPAL are presented in this section.

5.1 Parallelism

The P in OPAL stands for "parallelism". Therefore the new wakefunction must be parallel as well. A simple example is that cout does not work, *gmsg a OPAL function which provides that just processor 0 writes to the shell must be used instead. In OPAL the particles are already distributed over several processors. Using those local particles lead to a parallel implementation of adding the force to the particles. This could not be tested, since the function calling the wakefield is not yet parallel.

5.1.1 Adding the Force to the Particle

In OPAL the force due to the wakefields is not the only force acting on particles. There exist for example particle particle interaction, external electric and magnetic fields. Therefore one need to add the force due to wakefield to the already existing forces. Instead of having force as an variable opal store the E and B field at the particle and calculate the force using $F = q[\vec{E} + v \times \vec{B}]$. One can now add the force coming from the wake using $\vec{E} = \vec{E}_{coulomb} + \vec{E}_{extern} + \vec{E}_{wake}$. Where the electric field for the particle i is:

$$E(\vec{s(i)})_{wake} = \begin{pmatrix} E_x(s(i)) \\ E_y(s(i)) \\ E_z(s(i)) \end{pmatrix} = \begin{pmatrix} F_x(s(i))/q \\ F_y(s(i))/q \\ F_z(s(i))/q \end{pmatrix}. \tag{23}$$

5.2 Units

OPAL uses the SI units. In the Prototype some units were not in SI units for example the radius of the beam pipe was in mm instead of m. This had to be changed in the code to have a proper running simulation in OPAL.

5.3 One of the Challenges

Since OPAL is very powerful the amount of code already written is impressive. In the beginning it was challenging to cope with such a program. Even a easy thing like calculating the line density of the particle bunch, which is already provided as a function and therefore just can be called, was in the beginning demanding. Therefore a good documentation is needed to find those functions and to know how to apply them.

5.4 Documentation

A Doxygen documentation of the whole OPAL can be found at `http://amas.web.psi.ch`. There the documentation of the code written during the Bachelor thesis is integrated. As the C++ prototype and this code are documented with Doxygen the main part of the Doxygen comment used for the documentation could be reused from the prototype. Figure 8 shows where the function CalcWakeFFT() is documented.

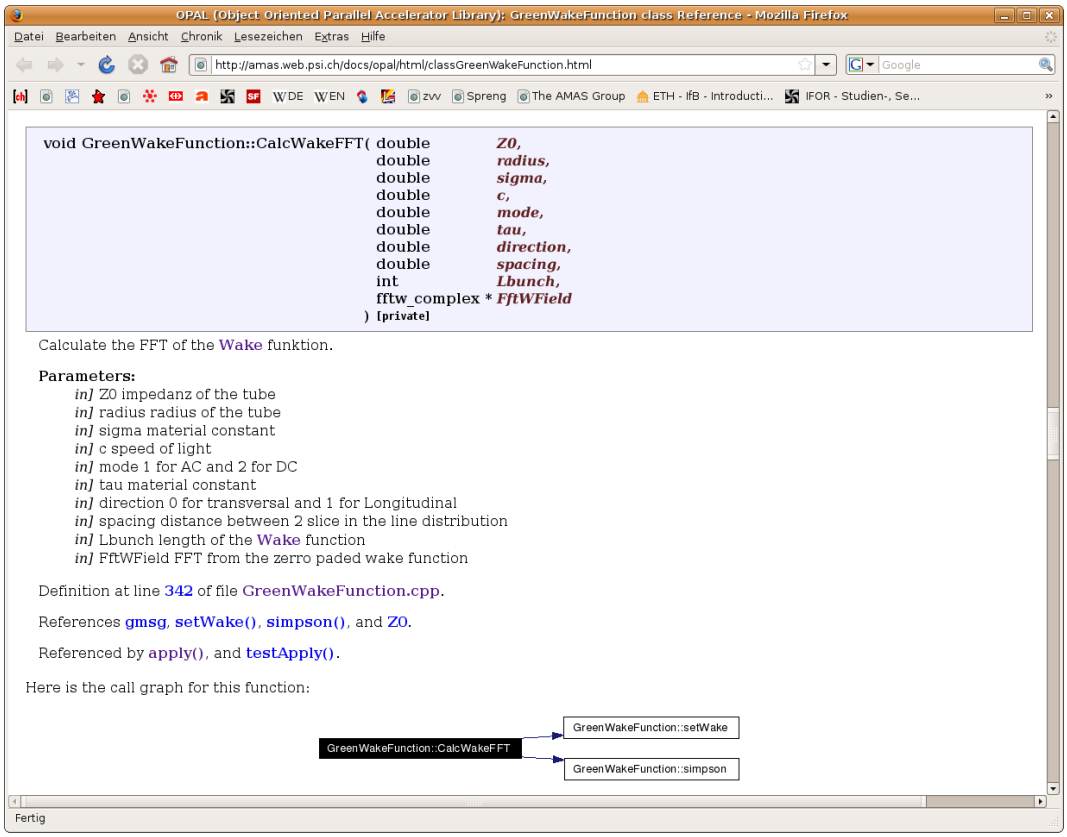

Figure 8: Example of the documentation of the code of this bachelor thesis in OPAL under `http://amas.web.psi.ch`

5.5 Parameters of the Wakefield

A wakefield in OPAL has several attributes. They are documented in the OPAL manual. By typing help,wake; in the running OPAL shell the attributes are printed on the screen. The result can be seen below:

```
The "WAKE" statement defines data for a the wakefunction on an element.
Attributes:
string   TYPE          Specifies the wake function: 1D-CSR,
                       LONG-SHORT-RANGE, TRANSV-SHORT-RANGE,
                       LONG-TRANSV-SHORT-RANGE
real     NBIN          Number of bins for the line density calculation
logical  CONST_LENGTH  True if the length of the Bunch is considered
                       as constant
string   CONDUCT       Cundivity: DC, AC
real     Z0            Impedanz of the beam pipe
string   FORM          The form of the  beam pipe: ROUND
real     RADIUS        The radius of the beam pipe [m]
real     SIGMA         Material constant dependant on the
                       beam pipe material
real     TAU           Material constant dependant on the
                       beam pipe material
```

5.6 Add Wakefunction to a Unit in OPAL

OPAL provides a meta language to simulate a accelerator. In this language few programming knowledge is needed to compose and simulate a hole accelerator. The next lines should give a rough idea how this meta language works and how easy a wake function can be added to an existing part of the accelerator. First one must specify the properties of the wake.

```
W1: Wake, TYPE="LONG-SHORT-RANGE", NBIN=10, CONST_LENGTH=true,
CONDUCT="AC", Z0=376.991118, FORM="ROUND",
RADIUS=0.005, SIGMA=6.45337e7, TAU=2.70187e-14;
```

Then one can add very easy a wake function to a existing part of the accelerator, in this case to a RF Cavity. In this to lines a RF Cavity with and one without Wake is shown.

```
FINLB01_RACF: RFCavity, L=0.54, VOLT=19.961*1.641907,
FMAPFN="FINLB01-RACF.T7", ELEMEDGE=0.069, TYPE="STANDING",
FREQ=1498.956, LAG=184.0/360.0, Wake=W1;

FINLB01_RACH: RFCavity, L=0.54, VOLT= 6.250*1.641907,
FMAPFN="FINLB01-RACH.T7", ELEMEDGE=0.069,
TYPE="STANDING", FREQ=4497.536, LAG=104.0/360.0;
```

The rest is the same as it would be with out wakefunction. Some of the following needed steps are shown below, to give a rough idea how easy a accelerator can be implemented. The particle beam must be specified, the single parts of the accelerator patched together, and then the beam can be tracked while going through the accelerator.

```
beam1: BEAM, PARTICLE=ELECTRON, pc=P0, NPART=5e4,
BFREQ=1498.953425154e6, BCURRENT=0.032912, CHARGE=-1;
l1: Line = (FINLB01_RACF, FINLB01_RACH);
track,line=l1, beam=beam1, MAXSTEPS=400, DT=1.0e-12;
```

6 Validation in OPAL

6.1 Comparison with the Prototype

6.1.1 Without SI Units

As mentioned in section 5.2 the Mathematica notebook [1] uses not SI units, for example mm instead of m. To avoid troubles with that, the first test in OPAL was made using the units from the Mathematica notebook. This first test should ensure that the part which is already implemented in the prototype works. Therefore the results of the wakefield code in OPAL was compared with the results from the prototype. To have comparable data the properties of the bunch must be simular. To ensure that, the length and the line density of the particle bunch were taken from the prototype and programmed in to a test function. Then the OPAL code could be tested under the same condition as the prototype. In Figure 9 and Figure 10 both simulations give the same results. That indicates that the calculation of the wakefield and of the force works in OPAL. The next step is to ensure that the simulation works correctly with the SI units used in OPAL.

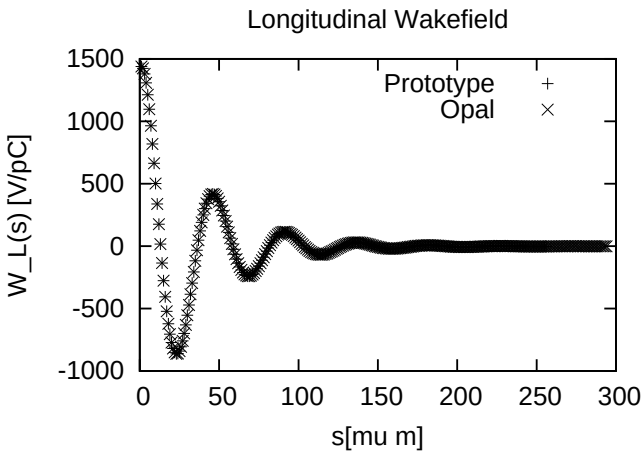

Figure 9: With equal properties the Simulation in OPAL gives the same wakefunction as the prototype. This is the longitudinal wakefunktion of a round, copper beam pipe with radius 5 mm and ac conductivity.

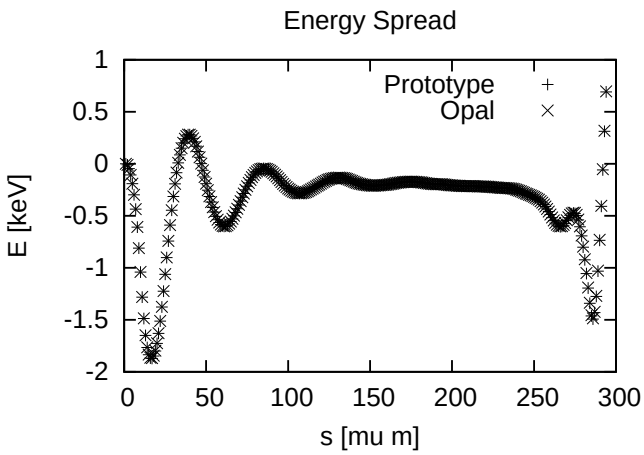

Figure 10: With equal properties the Simulation in OPAL gives the same kick as the prototype. This is the so called energy spread in [1] resulting from a longitudinal wakefunktion of a round, aluminum beam pipe with radius 5 mm and ac conductivity with the line density λ from [1].

6.1.2 With SI Units

All units are changed to SI units. To change the units is normally quite error-prone. Therefore this test is useful. The line density used in the prototype is still used in this test to compare the results with the prototype. This test shows that aside from a constant factor the results from Opal are equal to those of the prototype. To have good possibility of comparison this factor was calculate out of the results plotted in Figure 11. Therefore all the results of the wakefield calculated by OPAL were multiplied by a factor of 10. The as you can see on Figure 11 the results of the wakefield computed in OPAL give the same results as those from the prototype apart from the factor 10.

As for the wakefield the energy spread is also correct aside from a constant factor. Dividing all the results from OPAL by a constant factor of 10^6 the energy spread gives the same results as the prototype, as shown in Figure 12.

Unfortunately there is no time left in the my bachelor thesis to solve this problem. But i am confidently that the problem can be localized and solved quite fast.

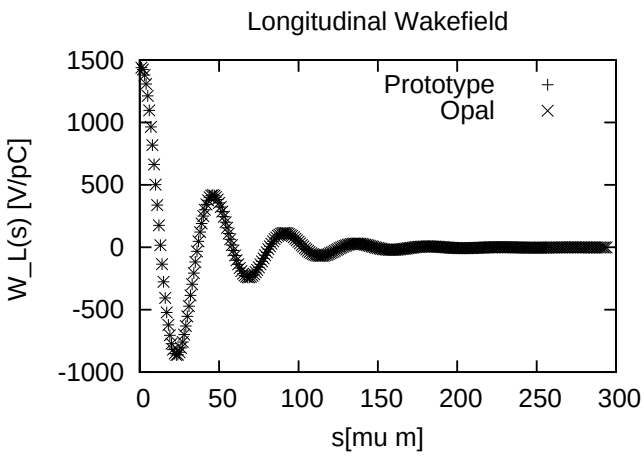

Figure 11: The wakefield of OPAL is multiplied by a factor of 10. Aside from this factor the OPAL code gives the same results as the prototype. This is the longitudinal wakefield of a round, copper beam pipe with radius 5 mm and ac conductivity.

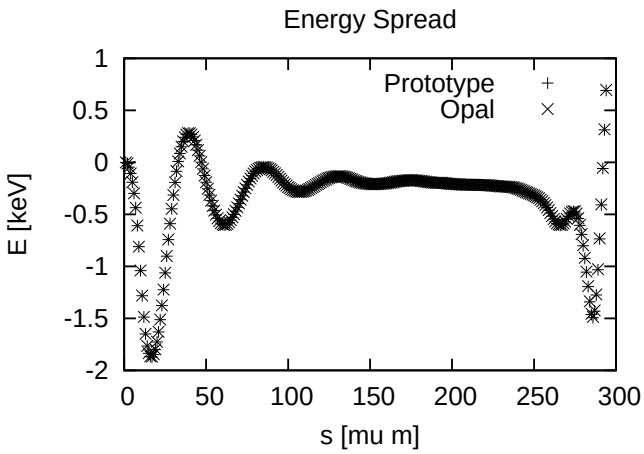

Figure 12: The energy of OPAL is divided by a factor of 10^6.Aside form this factor the OPAL code gives the same results as the prototype. This is the energy spread in [1] resulting from a longitudinal wakefield of a round, copper beam pipe with radius 5 mm and ac conductivity with the line density λ from [1].

7 Summary

Due to the good documentation of the wakefield implementation in OPAL it should be possible to continue the work started with this bachelor thesis. Possible next steps are presented in the following subsection.

7.1 Outlook

First the unsolved unit problem should be solved. Afterwards the code can be used for simulating the new XFEL at the PSI.

The calculation of the wakefield at the program start takes quite long. This part in not jet parallel. The parallelization of both equations (8) and (9) would be trivial. The sum over N could be calculated in parallel by p processors, in the consequence that each processor only make a sum over $\frac{N}{p}$. This calculation can be made without communication. Communication is only needed to gathered the result.

A speedup could also be managed if a schema with adaptive mesh would be used at the integration to get the wakefield.

Acknowledgment

I would like to thank Dr. Andreas Adelmann for the agreeable mentoring and the amas crew at the PSI for the interesting time I could spent with them at the PSI. Finally I would like to thank Prof. Dr. Peter Arbenz and Dr. Andreas Adelmann for giving me the chance to work on such an fascinating topic!

References

[1] C. Bontoiu and P. Craievich: How to use RW_wakefield notebook.

[2] A.W. Chao: Physics of collective beam instabilities in high energy accelerators, John Wiley & Sons, New York, 1993.

[3] J. Qiang: *IMPACT-T User Document Version 1.5*. Lawrence Berkeley National Laboratory, LBNL-62326, 2007.