The Henryk Niewodniczański
INSTITUTE OF NUCLEAR PHYSICS
Polish Academy of Sciences
ul. Radzikowskiego 152, 31-342 Kraków, Poland

www.ifj.edu.pl/publ/reports/2021/
Kraków, August 2021

Report No. 2106/AP

Extending of the *pytrip* software package for biologically optimized proton radiotherapy

Leszek Grzanka[*1,2], Kinga Jeleń[1], Łukasz Jeleń[1], Arkadiusz Ćwikła[1,2], Michał Krawczyk[2], Joanna Fortuna[1,2], Mateusz Łaszczyk[2], Paweł Olko[1]

1) Institute of Nuclear Physics, Polish Academy of Sciences, Kraków
2) AGH University of Science and Technology, Kraków

Abstract

Recent years brought several developments in treatment planning with variable radiobiological effectiveness for proton radiotherapy. Commercial clinical and research treatment planning systems are, in addition to their costs and availability, hard to adapt for emerging models and new approaches in proton therapy planning. We aim at providing suitable, open access and extensible software toolbox for research in treatment planning, based on the *TRiP98* program and freely available libraries.

The research platform is composed of two projects: a core library called *pytrip* and a graphical user interface (GUI) called *pytripgui*. Both projects are implemented mainly in the Python programming language. The core library is capable of handling DICOM and VOXELPLAN data format by custom readers based on the *NumPy* library. *Pytrip* provides a powerful toolbox for research in treatment planning. In our system, the treatment plan is calculated by the *TRiP98* program which can be run locally or remotely. Several additions not included in *TRiP98* are provided by the *pytrip* toolbox, like empirical models of variable RBE in proton therapy, calculations of DVH, DICOM support. Graphical user interface supplements *TRiP98* by a plotting tool, capable of handling CT scans and relevant quantities like dose, LET and RBE. GUI source code was upgraded to use a new widget toolkit - *PyQt5* framework, which enables maintenance and simplifies further development of the project.

Pytrip provides a powerful toolbox for research in treatment planning. It enables the development and testing of variable RBE models for proton therapy. Graphical user interface aids users in the visualization and calculation of treatment plans for research purposes.

[*] Corresponding author: Leszek.Grzanka@ifj.edu.pl

Wydano nakładem Instytutu Fizyki Jądrowej im. Henryka Niewodniczańskiego
Polskiej Akademii Nauk
Kraków 2021

ISBN 978-83-63542-23-8

Introduction

The background and objective of the project.
The area of biologically optimized treatment planning in proton therapy is attracting increasing attention due to potential improvements in therapy quality. Mathematical models of radiobiological effectiveness (RBE) [1,2] can be used to estimate the impact of variable RBE on treatment planning. Experimental evidence suggests that RBE is correlated with the linear energy transfer (LET) of the proton beam and the radiosensitivity of the biological endpoint (cell line or tissue). At the moment, the calculation of the variable RBE is not present in commercially available software for proton therapy treatment planning. Variable RBE is part of many research planning platforms, such as the RayStation research Treatment Planning System (TPS) [3] or the *TRiP98* platform [4].

This work describes the design and usage of *pytrip* software packages that extend the capabilities of the *TRiP98* research treatment planning system. The project was realized within the framework of the project Infrastructure in Proton International Research (INSPIRE) as a part of mathematical modelling and simulation work package (WP9). Major improvements within the scope of the work are the implementation of variable RBE models, improving support for handling DICOM data format and the complete redesign of the graphical user interface.

What is TRiP98?
TRiP98 (an acronym for TReatment planning for Particles, 1998 edition) is a research planning software dedicated to heavy-ion and proton radiotherapy [4]. It is a flexible tool that allows the testing of new models for the calculation of doses and optimization of treatment plans. *TRiP98* offers means to forward the calculation of spatial distribution for physical and biological doses, as well as for other parameters like linear energy transfer, RBE, alpha and beta parameters. The physical and biological dose distribution can be optimized by a dedicated module embedded in *TRiP98*. Biological dose optimization relies on the RBE tables, which can be generated by another software implementing the local effect model (LEM, distributed separately).

What is pytrip?
TRiP98 lacks a graphical user interface and is meant to be used as a command-line program on the Linux operating system. To expand the capabilities of the *TRiP98*, another project, named *pytrip*, was created [5]. Provides the means to use *TRiP98* functionalities (like dose optimization) in programs written in the Python programming language. Using the *pytrip* library greatly simplifies tasks in which treatment planning needs to repeat many times. Instead of manual execution, it is possible to use *pytrip* data structures for efficient batch processing. The *pytrip* library also extends *TRiP98* functionalities by providing empirical RBE models for proton radiotherapy: Wedenberg [1], McNamara [2], and Carabe [6].

What is pytripgui?
The *pytrip* core library is accompanied by a graphical user interface (GUI) called *pytripgui*. The purpose of the GUI is to minimize user interaction with the Python scripts and input text

files for the *TRiP98* command line program. GUI can be used for several tasks, including visualising patient CT data and precalculated dose distributions, preparation and execution of the optimisation process involving the *TRiP98* program. *Pytripgui* is intended to be used by users without any knowledge of programming or details of the *TRiP98* research platform.

Pytrip as an important tool for proton therapy studies
Pytrip was used as a crucial tool to evaluate the robustness of proton therapy with respect to interfractional motion. The study reported in [7] involved CT scans for 3 patients with pelvic lymph nodes (LN) irradiated from different beam angles. Single beam proton plans were prepared for all possible combinations of beam and couch angles (with 5^{o} intervals). A dedicated algorithm based on the *pytrip* software library was developed to account for the breathing cycle and to evaluate changes in water equivalent path length of proton beams. For all 3 patients, similar patterns of deterioration were found, with dose deficiency in the rectum, bladder, and normal tissues. Characteristic angles were found around 40^{o} and 150^{o} for left LNs and symmetrically for right LNs. These angles correspond to a treatment plan that is robust to dose deterioration in the target and organ at risk.

Pytrip is also a suitable tool for large-parameter studies. Ion beams that pass through heterogeneous materials, such as lung tissue, exhibit pronounced energy straggling. This phenomenon called the 'lung modulation effect' leads to a displaced and enlarged dose distribution of the pencil beam, deteriorating the dose distribution. A recent analysis [8] reveals that the lung modulation effect in proton and carbon ion radiotherapy could lead to an underdose in planned volume and overdose in healthy normal tissues. In that study, the authors developed a mathematical model and implemented it on top of the *pytrip* research treatment planning system. This model relied on a randomization process that required the recalculation of thousands of variations of the treatment plan. *Pytrip* enabled scripting this large parameter study and offered more advanced programmable manipulation of the plans that commercial systems offer.

Materials and Methods

Both the *pytrip* library and the *pytripgui* packages are implemented using the Python programming language. The main features of the *pytrip* library are:

- Object-oriented model for data structures used in treatment planning;
- DICOM support for CT images and data (dose, LET) cubes;
- DICOM support for structures (volume of interest);
- Volume Histogram calculations;
- Local and remote execution of TRiP98

The *pytrip* project - installation

To automatically download and install the *pytrip* library we use the standard pip package manager, included in most Python distributions. Installation can be done using a single command:

pip install pytrip98

Note that the installation package (*pytrip98*) is named differently from the library itself (*pytrip*). Once installed, the package can be imported from a Python command line or used in your Python program with import pytrip as pt. Downloading source code from the repository is recommended only for developers of the *pytrip* library. Proper installation from source code is a more demanding process as it requires manual installation of all dependent libraries and compilation of C extensions. The *TRiP98* program is not included in the *pytrip* library installation package and needs to be obtained and installed separately.

Architecture overview and example usage - *pytrip* project

Pytrip library offers several data structures (classes) needed to operate on input data (typically CT scans) and process data generated by optimization algorithms (i.e., spatial dose distribution). The top-level class is called Cube and is responsible for reading and writing data in VOXELPLAN format. It also holds most of the patient-specific data (i.e., patient name). For testing purposes, it also offers a way to create virtual patient structures, which can later be programmatically filled with data.
Derived classes include:

- CtxCube – responsible for computer tomography data
- DosCube – responsible for the spatial distribution of dose (physical or biological)
- LETCube – responsible for the spatial distribution of linear energy transfer (LET).

Each of these derived classes handles the reading and writing of data from the DICOM format. The DosCube class allows users to calculate and store dose-volume histogram (DVH) objects. Similarly, LET volume histograms can be calculated and saved using methods from the LETCube class. The inheritance diagram of these classes is shown in Figure 3.

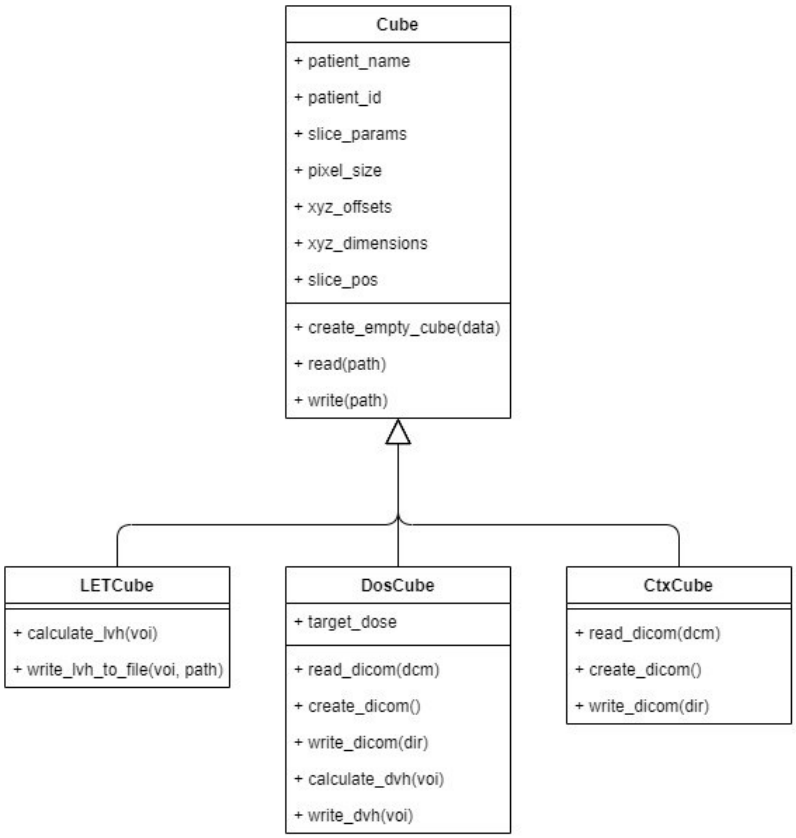


Figure 1 Diagram of inheritance for basic data structures in pytrip library code

Example usage:

```
import pytrip as pt

# Loading CT scan data from the directory with files in DICOM format
dcm = pt.dicomhelper.read_dicom_dir('path/to/dicom/directory')
c = pt.CtxCube()
c.read_dicom(dcm)

# Inspection of the CT cube, looking for minimum and maximum HU values.
print(c.cube.min(), c.cube.max())
```

The handling of treatment plans and the optimization process is governed by a set of classes. A class called Plan plays a central role as it stores information about single or multiple fields composing a treatment plan. Objects of Field class specify the beam and gantry angles, as well as projectile type: protons, carbon or other ions (via so-called kernel attribute). The irradiated volume is handled as an object of the Voi class, which supports reading data from the VOXELPLAN and DICOM standards. Each region of interest is divided into slices and further down into planar contours. The execution of the optimizing engine (*TRiP98* program) is governed by the Execute and ExecParser classes, which enable local or remote calculations. Relations between these classes are shown in Figure 2.

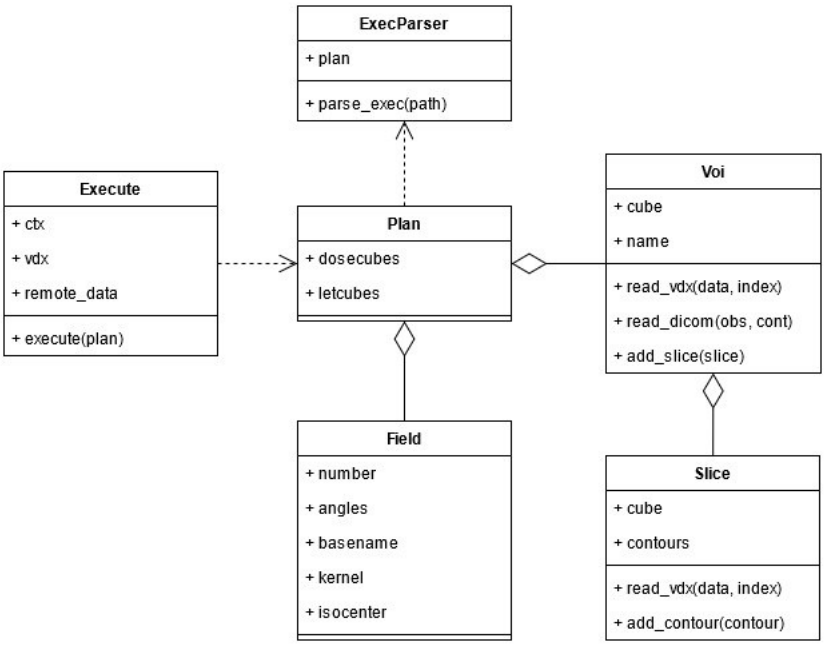


Figure 2 Diagram of relations between classes responsible for handling the treatment plan

As an example, we will use the following code to read CT data and patient contours defined in VOXELPLAN file format:

```
c = pt.CtxCube()
c.read(ctx_path)

v = pt.VdxCube(c)
v.read(vdx_path)
```

The following code will define projectile (here protons, in mykernel object), attach it to the irradiation plan, and finally define single field:

```
mykernel = pte.KernelModel()
mykernel.projectile = pte.Projectile("H", a=1)
mykernel.ddd_path = trip_path + "/DATA/DDD/1H/*"
mykernel.spc_path = trip_path + "/DATA/SPC/1H/*"
mykernel.sis_path = trip_path + "/DATA/SIS/19981218.sis"

plan = pte.Plan(basename=patient_name, default_kernel=mykernel)
plan.voi_target = v.get_voi_by_name('CTV')

field = pte.Field(kernel=mykernel)
field.gantry = 10.0 # degrees
field.couch = 90.0 # degrees
plan.fields.append(field)
plan.want_phys_dose = True
```

The following code will trigger the optimization process by executing the locally *TRiP98* program with an automatically generated configuration.

```
te = pte.Execute(c, v)
te.execute(plan)
```

We extend the *pytrip* core library by providing models of variable RBE. These models are implemented in the pytrip.models sub-package. It includes:

- Carabe, McNamara, and Wedenberg experimental models for proton therapy.
- the repairable-conditionally repairable (RCR) damage model [9] for carbon-ion radiotherapy

This package also includes the tumour control probability (TCP) model, based on [9]. The following code can be used to calculate the biological dose according to the Wedenberg model:

```
ab = 2.0 # assume that the alpha / beta ratio is equal to 2 Gy

# read CT data, dose and LET spatial distribution from precalculated values
c = pt.CtxCube()
c.read(ctx_path)

d = pt.DosCube(c)
d.read(dos_path)

l = pt.LETCube(c)
l.read(let_path)

# calculate biological dose as the product of dose cube and RBE cube
# Note that no loops are needed to perform multiplication
d_biol = d * rbe_wedenberg(dose=d, let=l, abx=ab)
```

Installation – *pytripgui* project

GUI can be installed similarly as the core library – using dedicated package manager pip:

```
pip install pytripgui
```

Users of the Windows operating system can also use a separate installer that can install GUI along with a Python programming environment.

Main features and architecture overview - *pytripgui* project

Main features of *pytripGUI*

- The interactive process of treatment planning
- Visualization of CT images, structures, and data cubes
- Support for various operating systems: Windows, Linux, and MacOS

PytripGUI is intended to be used as a standalone program installed by a user and operated as a graphical user interface.

Within this work, we organized the source code according to the Model-View-Component architectural pattern. The source code responsible for the presentation of data is

encapsulated into View classes (see Figure 4). The view layer relies on the PyQT5 Python library, which replaced wxwidgets used in the previous releases of *pytripGUI*.
The GUI is responsible for handling patient data and performing simulations with the optimized treatment plan (TRiP98). These tasks are handled by appropriate Executor and Model classes (see Figure 4).

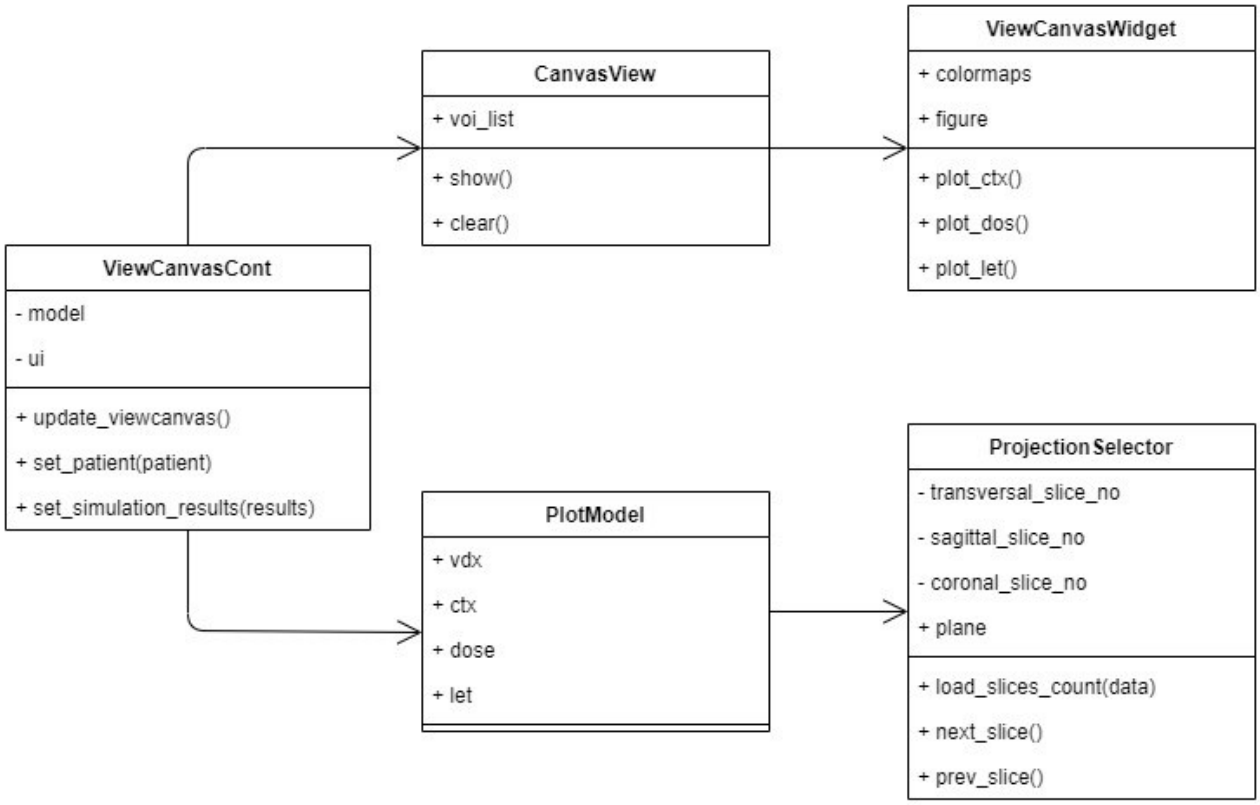


Figure 3 Diagram of relations between view and model classes in pytripGUI

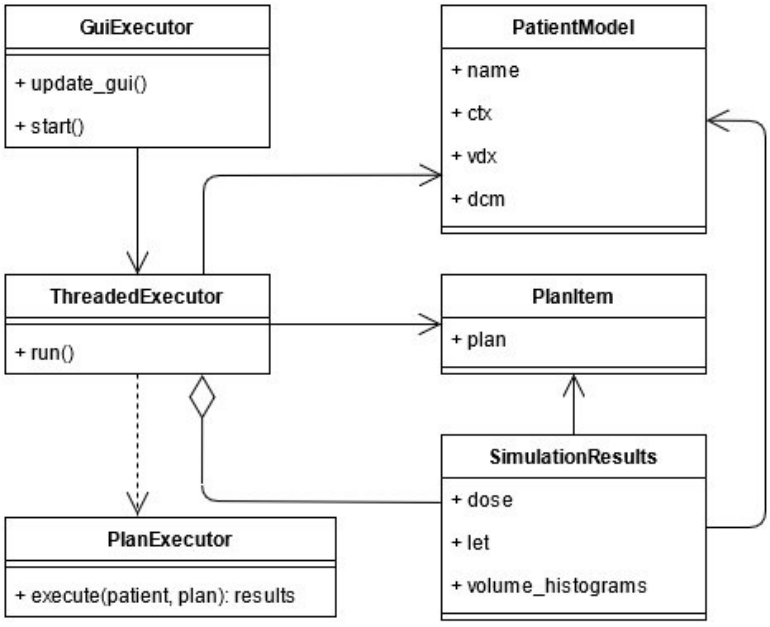
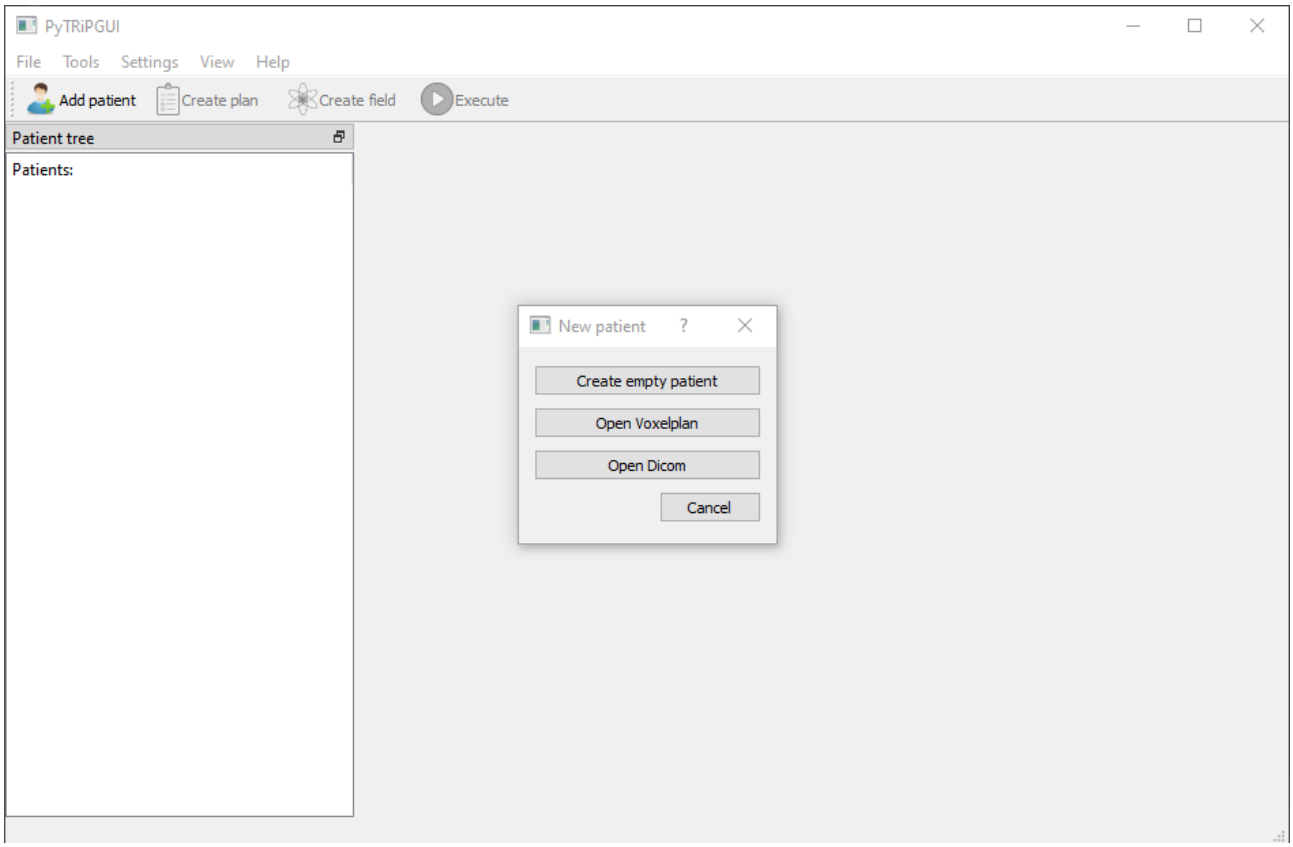


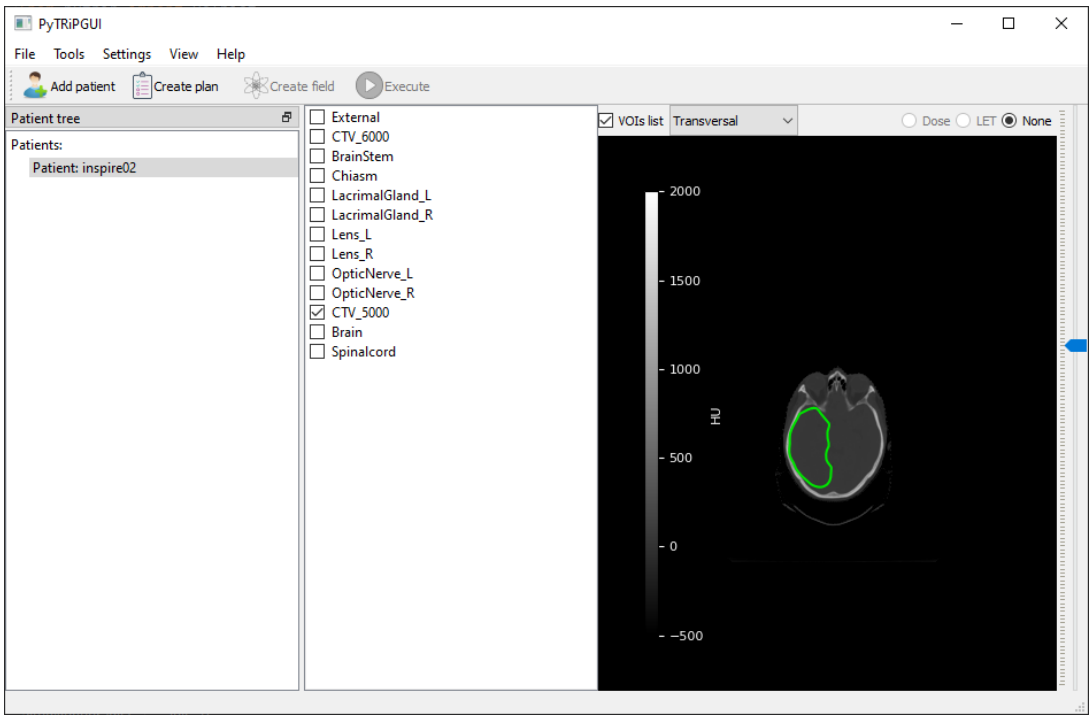
Figure 4 Diagram of relations between main classes of pytripGUI

User manual for working with *pytripgui*

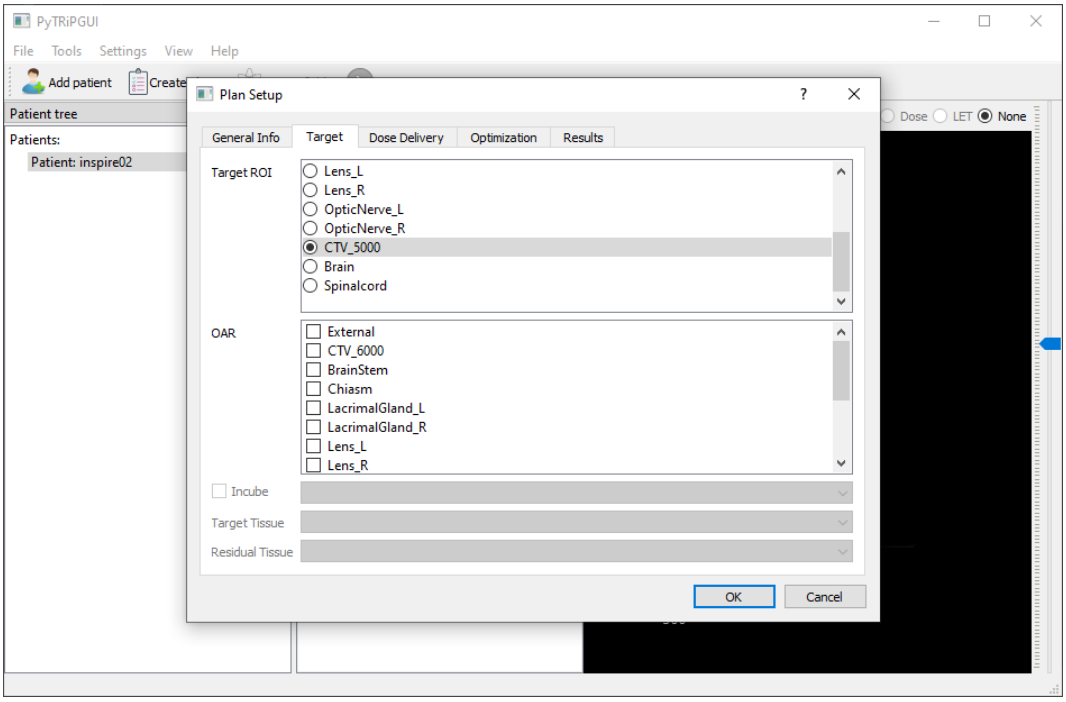
A typical session with *pytripGUI* starts with loading patient CT data (from VOXELPLAN or DICOM data formats). It is also possible to create virtual patient data for testing purposes.



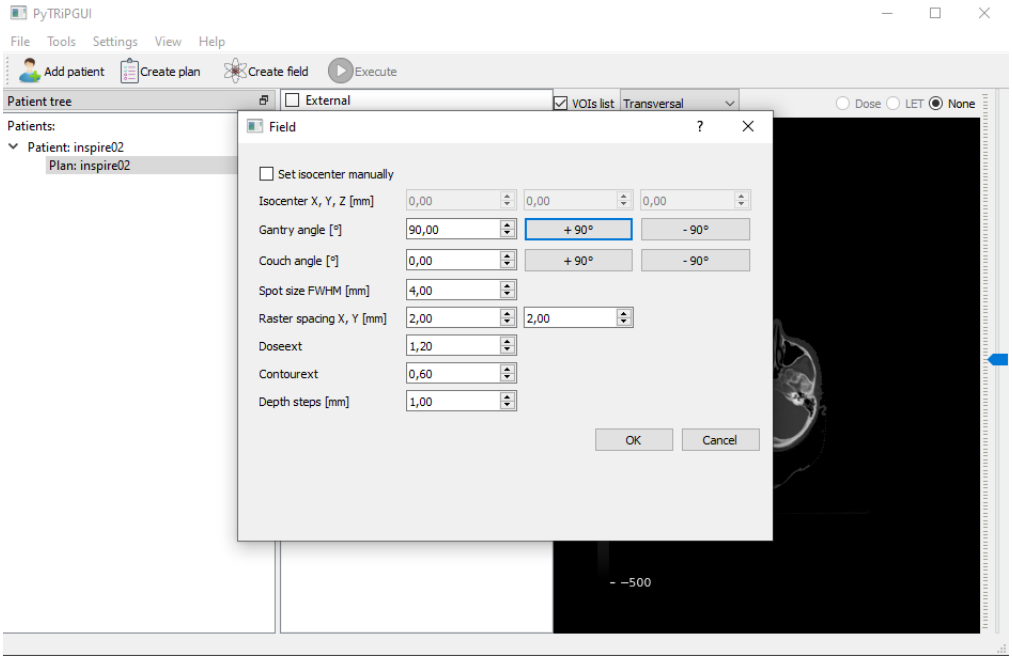
Once the patient CT data are loaded, the user can visualize transversal, coronal, and sagittal 2D sections through a CT scan. On the 2D sections, Hounsfield units are colour coded in greyscale. Contours of selected volumes of interest are also visible.



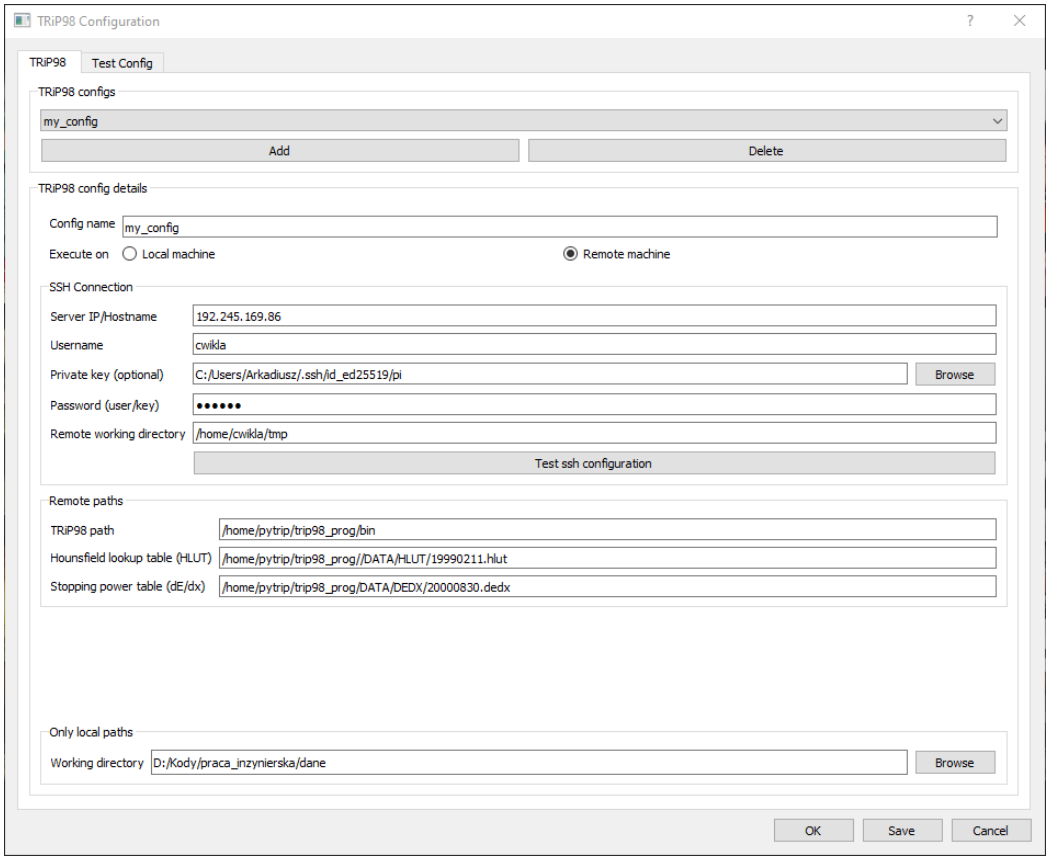
To proceed with the process of optimization of the treatment plan, the user selects the “Create Plan” option and chooses the target volume (so-called “Target ROI”). Several other options can be configured at the plan level, such as the simulation outcome (i.e., dose, LET, and RBE distributions).



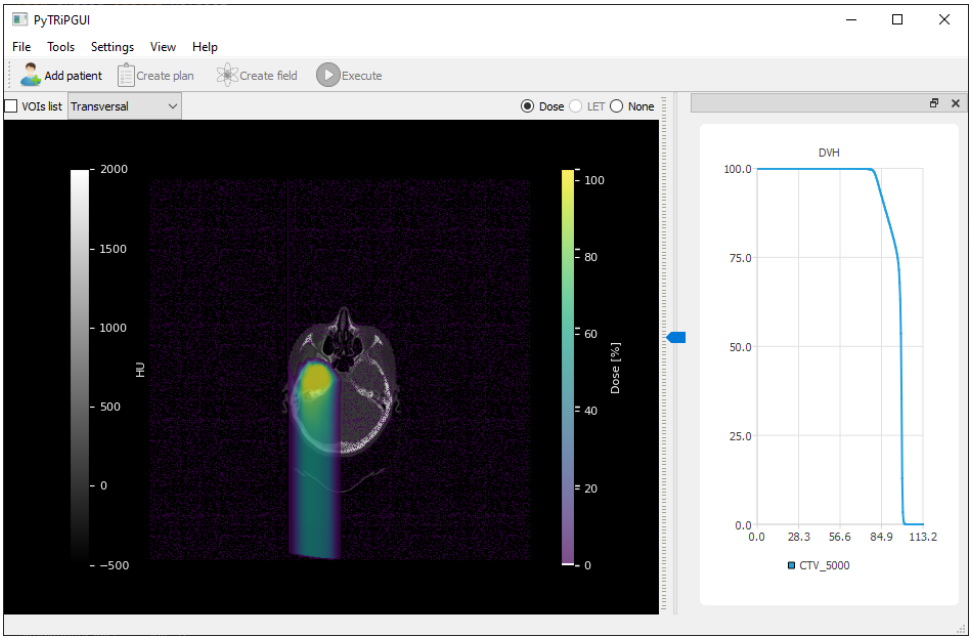
Once the plan is created, the user needs to specify one or more fields. At this level, the couch and gantry angles are specified.



Execution of the simulation can be local or remote. Both modes require the proper configuration of the TRiP98 optimisation engine. Crucial elements include beam models in the form of depth-dose distributions (DDD) and fragment-spectra (SPC) files containing depth-dose profiles and energy-fluence spectra.



Once the simulation is completed, the user can visualise the dose distribution presented on top of the anatomical structures in CT scans. In addition to the dose profile, the user can inspect the dose-volume histogram, as well as other plan metrics.



Project releases and development strategy

The development team uses a distributed version control system, git, to easily maintain code and independently add new features, make changes, and improvements in a developed application. We have chosen a popular provider: Github, which simplifies collaboration on the code between development team members and serves as a platform for task management. Two main repositories are used:

- Core library: https://github.com/pytrip/pytrip
- GUI: https://github.com/pytrip/pytripgui

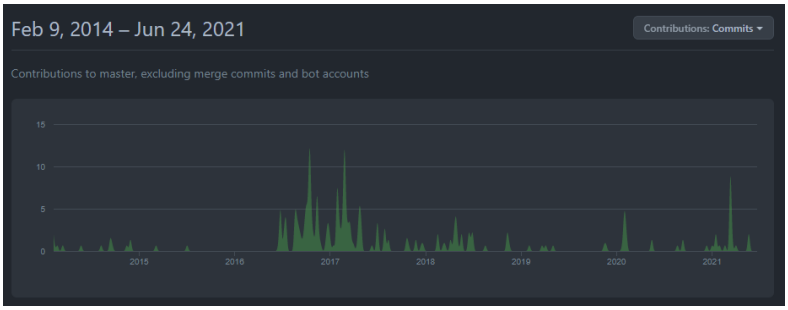


Figure 5 Activity on the GitHub repository storing source code of pytrip Python package. Data is recorded since 9.2.2014 when the code was migrated from the previous provider.

Additionally, Github allows some tests and checks to be run after each commit, which helps with maintaining the code clean and making sure it works properly. Both *pytrip* and *pytripgui* packages are being released after major changes in the code. At the time of writing last releases are the following:

- *pytrip* v.3.3.10 released on 18.08.2021
- *pytripgui* v1.2.0 released on 13.08.2021

Application of *pytrip* in variable RBE modelling for proton therapy

Robust treatment plans have low sensitivity to factors that can degrade the quality of the plan (i.e. configuration and range errors). Recent work [10] included first attempts to include RBE uncertainties in robustness evaluation. In our work [11], realised within the scope of the INSPIRE project we have extended the Wedenberg model to include a sophisticated model of error propagation starting from raw radiobiological data (cell survival curves). The original Wedenberg model has one free parameter, which is best fitted against experimental data to a single optimal value. By randomly sampling α and β parameters from joint distributions that reflect uncertainty and correlation in cell survival data, this single parameter can be replaced by a probability distribution. This, in turn, leads to a probabilistic version of the model from which a pseudo-random RBE can be sampled, exhibiting a skewed RBE distribution, as shown in Figure 6.

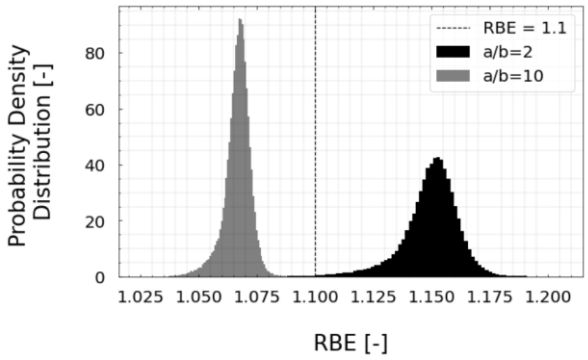


Figure 6. Probability distribution of RBE for two biological endpoints, characterized by alpha/beta ratio of 2Gy and 10Gy respectively.

The extended Wedenberg model was combined with the *pytrip* software package to calculate RBE uncertainties for monoenergetic proton beams (as shown in Figure 7**Błąd! Nie można odnaleźć źródła odwołania.**). The RBE uncertainty, defined as a function of mean to standard deviation ratio, reaches up to 2-3 % for the lowest proton energies (5

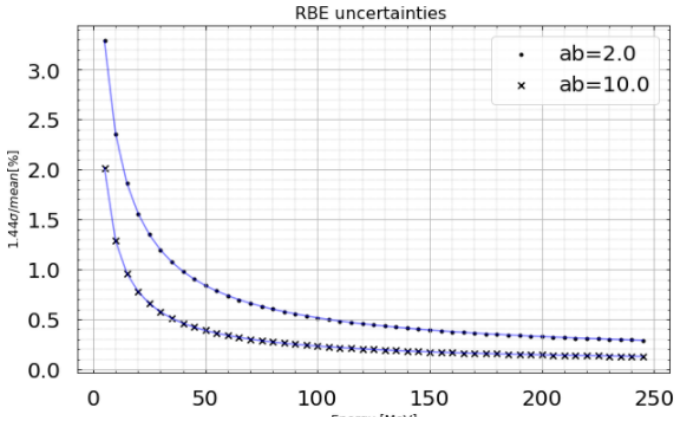


Figure 7 RBE uncertainty as the function of the primary proton energy for two biological endpoints

MeV). These values are of similar magnitude as the dose and position uncertainties but should be included together in the uncertainty budget calculations.

To assess the impact of RBE uncertainty on plan robustness we prepared a treatment plan optimised for constant physical dose using the *pytrip* project. The plan was based on

anonymised patient data with prostate tumour, assuming pencil beam scanning irradiation with 3 fields. The extended Wedenberg model was used to calculate the RBE distribution in each voxel. Batch processing capabilities of the *pytrip* package enabled us to obtain large statistics (10000 samples) and extraction of mean RBE and biological dose values for voxel. A single slice in the transverse plane (as shown in Figure 8) demonstrates the distribution of biological dose with this approach. Elevated RBE values with higher uncertainty are present in the distal part of the treated volume and indicate the region for which dedicated studies are needed to assess its impact on the plan quality.

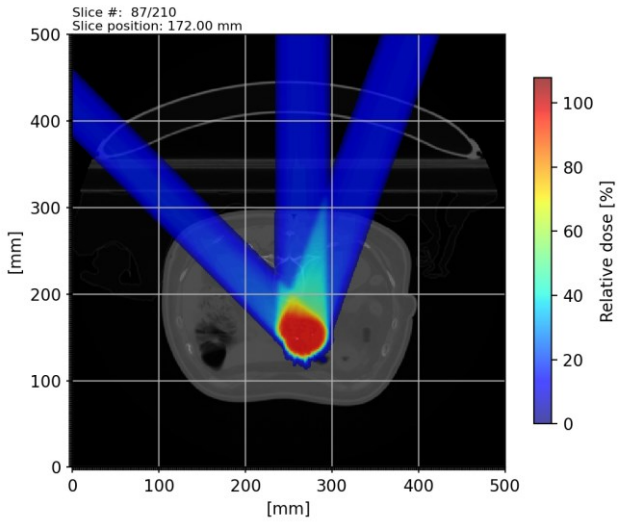


Figure 8 Spatial distribution of biological dose, calculated according to extended Wedenberg model for a patient with prostate tumor.

Conclusions

We have extended the capabilities of the *pytrip* software packages by including models of variable RBE. The graphical user interface was renewed and migrated to currently supported programming libraries. *Pytrip* now targets two groups of users: experienced programmers, using scripting languages for data analysis, and regular users who need a user-friendly graphical user interface to work with.

Pytrip has already been used as a powerful tool for parametric study, enabling research to simulate batches of thousands of treatment plans and control the programmatic treatment planning process. It was also an essential tool in developing a probabilistic version of the Wedenberg model of variable RBE. This model was then used in studies of the robustness of the plan under uncertainties derived from radiobiological data.

Acknowledgements

This publication has been prepared to report the progress of the work achieved within the Horizon 2020 project INSPIRE Grant 730983 and was partly financed from the project. This research was also supported in part by PL-Grid Infrastructure. We are grateful to N. Bassler for his collaboration on this report and inspiring discussion.

References

[1] M. Wedenberg, B.K. Lind, B. Hårdemark, A model for the relative biological effectiveness of protons: the tissue specific parameter α/β of photons is a predictor for the sensitivity to LET changes, Acta Oncol. 52 (2013) 580–588. https://doi.org/10.3109/0284186X.2012.705892.

[2] A.L. McNamara, J. Schuemann, H. Paganetti, A phenomenological relative biological effectiveness (RBE) model for proton therapy based on all publishedin vitrocell survival data, Physics in Medicine and Biology. 60 (2015) 8399–8416. https://doi.org/10.1088/0031-9155/60/21/8399.

[3] J. Ödén, K. Eriksson, I. Toma-Dasu, Inclusion of a variable RBE into proton and photon plan comparison for various fractionation schedules in prostate radiation therapy, Med. Phys. 44 (2017) 810–822. https://doi.org/10.1002/mp.12117.

[4] M. Krämer, Swift ions in radiotherapy – Treatment planning with TRiP98, Nucl. Instrum. Methods Phys. Res. B. 267 (2009) 989–992. https://doi.org/10.1016/j.nimb.2009.02.015.

[5] J. Toftegaard, J.B. Petersen, N. Bassler, PyTRiP - a toolbox and GUI for the proton/ion therapy planning system TRiP, J. Phys. Conf. Ser. 489 (2014) 012045. https://doi.org/10.1088/1742-6596/489/1/012045.

[6] A. Carabe, M. Moteabbed, N. Depauw, J. Schuemann, H. Paganetti, Range uncertainty in proton therapy due to variable biological effectiveness, Phys. Med. Biol. 57 (2012) 1159–1172. https://doi.org/10.1088/0031-9155/57/5/1159.

[7] O. Casares-Magaz, J. Toftegaard, L.P. Muren, J.F. Kallehauge, N. Bassler, P.R. Poulsen, J.B.B. Petersen, A method for selection of beam angles robust to intra-fractional motion in proton therapy of lung cancer, Acta Oncol. 53 (2014) 1058–1063. https://doi.org/10.3109/0284186X.2014.927586.

[8] T.P. Ringbæk, A. Santiago, L. Grzanka, K. Baumann, V. Flatten, R. Engenhart-Cabillic, N. Bassler, K. Zink, U. Weber, Calculation of the Beam-Modulation Effect of the Lung in Carbon Ion and Proton Therapy With Deterministic Pencil Beam Algorithms, Frontiers in Physics. 8 (2020) 520. https://doi.org/10.3389/fphy.2020.568176.

[9] L. Antonovic, E. Lindblom, A. Dasu, N. Bassler, Y. Furusawa, I. Toma-Dasu, Clinical oxygen enhancement ratio of tumors in carbon ion radiotherapy: the influence of local oxygenation changes, J. Radiat. Res. 55 (2014) 902–911. https://doi.org/10.1093/jrr/rru020.

[10] J. Ödén, K. Eriksson, I. Toma-Dasu, Incorporation of relative biological effectiveness uncertainties into proton plan robustness evaluation, Acta Oncol. 56 (2017) 769–778. https://doi.org/10.1080/0284186X.2017.1290825.

[11] K. Jeleń, L. Grzanka, P. Olko, UNCERTAINTY OF RBE MODEL IN PROTON RADIOTHERAPY BASED ON α/β RATIO AND LINEAR ENERGY TRANSFER, Acta Phys. Pol. B. 51 (2020).