A Prostate Cancer Computer Aided Diagnosis Software including Malignancy Tumor Probabilistic Classification

A Prostate Cancer Computer Aided Diagnosis Software including Malignancy Tumor Probabilistic Classification

Keywords: Prostate Cancer, Computer Aided Diagnosis, Tumor Probabilistic Classification, Software Design

Abstract: Prostate Cancer (PCa) is the most common solid neoplasm in males and a major cause of cancer-related death. Screening based on Prostate specific antigen (PSA) reduces the rate of death by 31%, but it is associated with a high risk of over-diagnosis and over-treatment. Prostate Magnetic Resonance Imaging (MRI) has the potential to improve the specificity of PSA-based screening scenarios as a non-invasive detection tool. Research community effort has been put on classification techniques based on MRI in order to produce a malignancy likelihood map. The paper describes a Computer Aided Diagnosis (CAD) software, still under development, which aims at providing a comprehensive diagnostic tool that include an integrated classification stack, from a preliminary registration of images of different sequences to the classification process. The description includes the prototyping design, the implemented work-flow and the software architecture. This software can improve the diagnostic accuracy of the radiologist, reduce reader variability and speed up the whole diagnostic work-up.

1 INTRODUCTION

Prostate cancer (PCa) is the most common malignancy affecting men in the world. It represents the third cause of cancer death in industrialized countries (Ferlay et al., 2013). Due to the widespread use of screening tests, such as Prostate Specific Antigen (PSA), five-year survival rate is high. Still PCa diagnosis is obtained by transrectal ultrasound (TRUS) guided core biopsy: several samples, usually from 12 to 16, are taken from different parts of the gland in order to increase the probability to detect cancer. Currently, there are several limitations to the applied diagnostic-therapeutic work-flow, which negatively affect the quality of life of the subjects and/or may lead to over or under-treatment of PCa (Lujan et al., 2004). Limitations can be summarized as follows:

- PSA test has a low specificity in detecting PCa, and can cause potential harms: additional medical visits, adverse effects of prostate biopsies, anxiety, and over-diagnosis leading to over-treatment with its associated side effects (bowel urgency, urinary leakage, erectile dysfunction).
- Whenever PCa is suspected, patients undergo biopsy guided by TRUS, having a low detection rate and low-specificity. This may determine a diagnostic delay, which leads to a higher rate of recurrence and to a lower patient survival rate.
- Because of the lack of techniques for precise PCa localization and staging, current treatment strategies involve the whole gland. Due to the inherent risks associated with surgical resection and radiotherapy, many patients develop severe side effects.

Magnetic Resonance Imaging (MRI) has been shown promising in the identification of PCa. Physicians commonly exploit it for preoperative evaluation, cancer staging, and image guidance for prostate interventions (Türkbey et al., 2012).

In particular, combining different MRI sequences, it is possible to derive anatomical and functional information useful to provide not only tumor localization but also to distinguish low risk tumors (amenable to a wait-and-see strategy or to active surveillance) from high-risk patients that will require prostatectomy. In those terms, the combining process, the so called multi-parametric (mp) MRI, brings detection rates of clinically significant lesions consistently up to 90% (Liu et al., 2009; Hoeks et al., 2011). However, the more variables are introduced the more difficult it is, even for the experienced reader, to integrate the available information into one reliable final report. Usually, the interpretation of a prostate MRI examination requires dedicated radiologist. It is a time consuming task, which still can lead to loose important information due to the complexity of comparing high resolution anatomical features of the T2-weighted (T2-w) images, with functional data originating from dynamic contrast enhanced (DCE) studies and diffusion weighted (DW) imaging. The complexity has been observed in several studies as high inter-reader variability (Dickinson et al., 2011).

Computer Aided Diagnosis (CAD) softwares can help the radiologist in the diagnostic work-up and speed up reporting tasks by automating the mp MRI process from the different MRI sequences (Padhani et al., 2011; Hegde et al., 2013). Their development is speeding up due to the need of less invasive medical procedures, and MRI analysis has proven to be effective in tumor diagnosis [Citations Omitted For Blind Review]. Most of actual CAD solutions are available as commercial products, like the DynaCAD by Invivo (Invivo, 2013), so they are not designed and developed to be integrated with new features. Moreover, most of them comes with the hardware for the MRI.

On the open source side, a wide community of developers has been involved in several open source projects to develop libraries, development frameworks and software tools to the community itself. Some of them, like the *3DSlicer* framework (3DSlicer, 2013), are very popular among the researchers community because they provide lots of algorithms and filters to easily manipulate medical images (Pieper et al., 2006; Hegde et al., 2013; Penzkofer and Tempany-Afdhal, 2013). Unfortunately, they can easily help on research tasks but they lack of simplicity when the final goal is building a CAD software with a custom Graphical User Interface (GUI). In fact, very effective frameworks, such as 3DSlicer, that provide GUI capability, are not meant to build that GUI from scratch (by forcing the user to interact with a *standardized* interface). On the other hand, looking for a modular and flexible software architecture design and implementation, which requires to plug new features in when a new algorithm or a feature will be proposed, introduce a challenging complexity in the software design that can take advantage from framework and libraries designed for that purpose. Finally, all available tools are not designed with a clinical work-flow as final development stage in mind.

In this paper, we present a CAD software currently under development by two research institutions [Omitted For Blind Review].The idea is to develop a complete tool for assisted diagnosis, from the patient data management to the final medical report, that includes a probabilistic malignancy classification of tumor tissue by imaging. The tool meets all modern software paradigms, in terms of multi-tasking and multi-platform solutions. By overcoming the limitations of the current subjective way of analyzing MR data, this approach could substantially improve the diagnostic and therapeutic work-up of individuals with PCa, and in the end improve quality of patients life dramatically.

The paper describes the tool by describing the work-flow designed for the software (Section 2). Section 3 introduces the software architecture design. Eventually, Section 4 gives some conclusion and future perspectives to the tool development.

2 THE SOFTWARE WORK-FLOW

The idea of our CAD software was to design a tool to support the diagnosis process by providing a built-in MRI multi-volumes view, including a malignancy probability map. It has to implement a comprehensive work-flow able to guide the physician from image upload to the final medical report.

The software work-flow design requires to simplify the user experience, by drawing sequences of operations that quickly reach what the user is expected to do. Design very complex work-flow leads to a huge variety of behaviors the software is going to be able to manage. This huge variety commonly comes with low usability of the software due to the difficulty to prevent all possible combinations of those behaviors. General purpose software may take advantage of a so wide flexibility given by lots of possible combinations of operations, but specialistic software do not. In our case, we have to keep the user focused on the main tasks the software is designed for.

In particular, the classification process requires a deep analysis to be included in the software design, in particular, by identifying the correct way to integrate it in the work-flow and by designing the proper user interaction with the classification output. The final work-flow is arranged to follow only two different paths:

1. The patient management, where the user has either to import new patients to be diagnosed or to manage already in-platform ones.
2. The diagnosis work-up, where the physician has to be guided from the patient analysis (through the visual analysis of MRI volumes and classification results) to the patient medical report (useful either for further analyses or for pre-surgical reports).

This entire work-flow is shown in Figure 1.

The patient management path is the most simple, and can be split in two possible flows: (i) the patient import and (ii) the already in-platform patients management. We start focusing on the first one.

Importing a patient means, at first, being able to analyze a MRI folder, which contains all image volumes generated by the MRI machine for that patient exam, and identify all available volumes, providing the user with their full list. Then, the user can choose

the set of volumes to be imported so the import process starts. During the import process, all imaging analysis algorithms run, including registration stages, multi-parametric analysis and the classification process [Omitted For Blind Review], and the physician has to enter all information about the patient (e.g., blood test results and so on). Only when all the analyses are completed the user will be able to conclude the import.

The management of patients in-platform allows to modify all previously entered data and to add a new MRI exam for a given patient. In this case, the workflow is an alternative case of the import one: only the MRI folder has to be analyzed and patient information is available to be modified (e.g., if the new exam comes with new blood tests).

The diagnosis work-up is the most flexible and the most strict at the same time. The classification process requires a deep analysis to be included in the software design, in particular, by identifying the correct way to integrate it in the work-flow and by designing the proper user interaction with the classification output.Once a patient is selected, the radiologist can select the MRI volumes to be displayed, together with the probability map superimposed to the T2-w axial MRI image (see Figure 2). Then the workflow lets the physician analyze all data, by navigating volumes, i.e., in single or multi view mode. In the multi-view mode, the diagnosis is aided by accessing up to 4 views simultaneously, which may include any of the performed MR sequences, such as DCE volumes, DWI volumes and so on. Basic image navigation tools like zoom, pan and re-slice need to be available for each volume and time-related volumes (such as DCE) navigation may be either space or time dependent.

During the diagnosis, the user is supported by a set of image interacting tools, such as define hand-free and circular Region of Interests (ROIs), and a set of extra elaborations among images, such as retrieve wash-in and wash-out plots of ROIs. Physicians may add annotations free-text or predefined text related to one of the 16 prostatic regions of interest, as previously reported (Dickinson et al., 2011), and create screenshots of peculiar image areas (see Figure 3).

Once the physician analysis is completed, a report form will be available to provide the final medical report. The form is provided by a set of common word-processor tools to write the report part. All previous annotations and screenshots will be also reachable to be linked in the report, and an interactive 16 prostatic regions of interest map allows the physician to better annotate the tumor morphological position. The report is included in the data stored for each patient.

3 THE SOFTWARE ARCHITECTURE

The software architecture has been designed to rely on two main levels: the internal core engine and the visualization engine (Figure 4). The internal core engine implements all MR images manipulation, classification and data analysis processes. The visualization engine defines all graphic aspects from the GUI interaction to the graphical rendering of MRI volumes. We chose to implement the internal core engine using Insight Toolkit (ITK) (Kitware, 2013a) and Boost (Dawes et al., 2013) libraries, while the visualization engine is QT-based (Digia plc, 2013) application supported by the Visualization Toolkit (VTK) library (Kitware, 2013b). They have been chosen because they all are multi-platform libraries. Both engines are based on a pipeline execution paradigm inherited from ITK and VTK principles (Pieper et al., 2006).

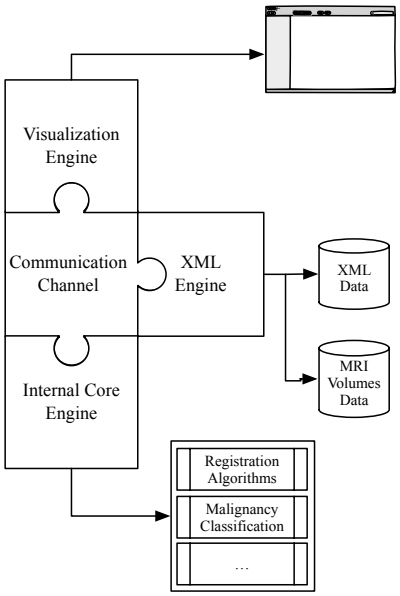

Figure 4: The Software Architecture exploited.

The software can be considered as an on-going research tool so the internal core engine implementation, whenever possible, is modular. This way new algorithms can be developed and plugged-in without rewriting the software. The modular approach is typical of development framework like 3DSlicer, where it has been reached by building an interpreter engine above the *real* execution engine. Due to the interpretation step required, performances may suffer of this kind of architectural implementation.

We chose to resort to a modular approach exploiting the C++ template mechanism. We designed all

core processes, such as the classification process, as *template*-based processes: the implementation of new algorithms will respond to their compliance with the template paradigm. In particular, for the classification part of the internal core engine, which is the most innovative part of the software, we carefully defined common input and output formats as guideline for new algorithms. This way, changes in the classification strategy can be implemented by new modules without requiring changes in the execution pipeline. In particular, all registration algorithms involved in the patient analysis are also modular, and they can be updated or modified with the same approach.

It can be said that modern algorithms, especially biological and imaging ones, require a high parallel implementation in order to cope with complex tasks and to be solved in a reasonable time (Tuckerman et al., 2000; Glockner et al., 2005). The imaging and the classification algorithms heavily suffer of time consuming issues that have been addressed by all modern computational techniques to speed them up, such as multi-core or multi-GPUs implementations (Baskaran et al., 2010; Vasanawala et al., 2011). Even so, the user interaction itself requires the software to be ever responsive (Betz et al., 2000) and to do so the implementation is based on parallel tasks. Using Boost and ITK multi-threading capabilities, we have been able to provide the internal core engine of parallel architecture and implementation. Resorting to QT, ITK and VTK event management paradigm, we design an event-based communication channel to connect the two cores. All parallel tasks will be synchronized using the communication channel: once an internal core engine task starts, i.e. an imaging algorithm, the visualization engine is still able to provide the user a continuous interaction. Once the task emits a completion event, the communication channel takes care of the event and forwards it to the visualization engine that can behaves as needed. So the communication channel allows the responsiveness of the GUI by including in the software architecture an intermediate level that takes care of all time consuming algorithms.

A final aspect we define in the architecture is the data infrastructure and management. Since the amount of data required in a diagnostic tool is a huge one (tons of images and patient information), data have to be manipulated with ease. Because the development of a full-feature version of the software may take lots of time, the data have to be modified also outside the software, i.e. by hand, both in terms of schema and information (e.g., a new patient added, a new set of information previously missing). Moreover, most of the data is a sensible one, so the data management has to provide anonymizing features. At this point, we mainly have two different choices: (i) a relational database management system (RDBMS) or (ii) an XML data management system.

XML and RDBMS are often mixed up together (Paoin and Boonchai-Apisit, 2009; Taghva and Jayakumar, 2009) because XML allows to easily exchange data or to define very complex queries (more than SQL does in some contexts, e.g., bibliography searching) whereas relational databases organize, store and access data in a more efficient way. Providing our software to a RDBMS would require including in the software architecture an external engine to manage a small part of the data: the patient information. In fact, all imaging-related data, such as MRI volumes, is not conveniently stored in a relational database. Nevertheless, RDBMS data do not come in a human-readable (due to data compression and indexes organization), so it ever requires an intermediate level of translation to be modified by the user. Instead, XML data is a more human-readable format, which can easily handle all patient data and still permits a direct modification, even outside the tool. From the architectural point of view, XML engines are usually software classes that can easily be included in a software project. In fact, QT contains several different libraries to manipulate XML-based documents. Thus, we chose to add a XML engine to manipulate an XML-based database. Since the XML engine is implemented in QT, it can be plugged in the communication channel without problem.

Confidentiality has been reached by providing the engine of an encryption module, leaving the possibility of further adding different encryption algorithms, and by properly designing the GUI not to show the patient's sensible information. Eventually, the XML engine should also implement error handling to avoid difficulties during the user testing phase of the project. Whenever a physician involved in that phase will corrupt the data for any reason, it should be easy to solve the problem.

It can be seen that, at architectural level, we reached a very flexible organization, taking care of further implementation of new functionalities that may arise from the research field.

4 CONCLUSIONS

In this paper we give a comprehensive description of the tool we are working on. The methodological approach for all the design aspects has been exploited in order to give a better insight of the tool functionalities. Although the software is under development and still

not available outside [Omitted For Blind Review], the first versions and demos show a promising scenario. As soon as the main functionality will be considered as stable, further versions will see the preliminary in-field test, allowing the use in the diagnostic work-up. At that development stage we plan to have a big milestone review of the project, including physicians feedbacks and intensive multi-platform performances test.

The software architecture designed guarantees that the software lifetime can be extended by easily adding new modules and partially re-programming the internal core pipeline if needed. The prototyping approach is going to be the project guideline for further CAD project, contributing to help time and cost reduction.

Eventually, once we'll reach a proper development stage, we plan to further evaluate all feasible contributions to the ITK and VTK libraries in terms of modules to be proposed to the community.

REFERENCES

3DSlicer (2013). 3DSlicer Project. http://www.slicer.org.

Baskaran, M., Ramanujam, J., and Sadayappan, P. (2010). Automatic c-to-cuda code generation for affine programs. In Gupta, R., editor, *Compiler Construction*, volume 6011 of *Lecture Notes in Computer Science*, pages 244–263. Springer Berlin Heidelberg.

Betz, K., Leff, A., and Rayfield, J. (2000). Developing highly-responsive user interfaces with dhtml and servlets. In *Performance, Computing, and Communications Conference, 2000. IPCCC '00. Conference Proceeding of the IEEE International*, pages 437–443.

Dawes, B., Abrahams, D., and Rivera, R. (2013). Boost c++ libraries. [Available Online]: http://www.boost.org/doc/libs/.

Dickinson, L., Ahmed, H. U., Allen, C., Barentsz, J. O., Carey, B., Futterer, J. J., Heijmink, S. W., Hoskin, P. J., Kirkham, A., Padhani, A. R., Persad, R., Puech, P., Punwani, S., Sohaib, A. S., Tombal, B., Villers, A., van der Meulen, J., and Emberton, M. (2011). Magnetic resonance imaging for the detection, localisation, and characterisation of prostate cancer: Recommendations from a european consensus meeting. *European Urology*, 59(4):477 – 494.

Digia plc (2013). QT Project. http://qt-project.org/.

Ferlay, J., Steliarova-Foucher, E., Lortet-Tieulent, J., Rosso, S., Coebergh, J., Comber, H., Forman, D., and Bray, F. (2013). Cancer incidence and mortality patterns in europe: Estimates for 40 countries in 2012. *European Journal of Cancer*, 49(6):1374 – 1403.

Glockner, J. F., Hu, H. H., Stanley, D. W., Angelos, L., and King, K. (2005). Parallel mr imaging: A user's guide1. *Radiographics*, 25(5):1279–1297.

Hegde, J., Mulkern, R., Panych, L., Fennessy, F., Fedorov, A., Maier, S., and Tempany, C. (2013). Multiparametric mri of prostate cancer: An update on state-of-the-art techniques and their performance in detecting and localizing prostate cancer. *J Magn Reson Imaging*, 37(5):1035–1054.

Hoeks, C. M. A., Barentsz, J. O., Hambrock, T., Yakar, D., Somford, D. M., Heijmink, S. W. T. P. J., Scheenen, T. W. J., Vos, P. C., Huisman, H., van Oort, I. M., Witjes, J. A., Heerschap, A., and Fütterer, J. J. (2011). Prostate cancer: Multiparametric mr imaging for detection, localization, and staging. *Radiology*, 261(1):46–66.

Invivo (2013). DynaCAD by Invivo Coorporation. http://www.invivocorp.com/avs/prostate.php.

Kitware (2013a). ITK Project. http://www.itk.org/ITK/project/project.html.

Kitware (2013b). Visualization Toolkit (VTK) Project. http://www.vtk.org/.

Liu, X., Langer, D., Haider, M., Yang, Y., Wernick, M., and Yetik, I. (2009). Prostate cancer segmentation with simultaneous estimation of markov random field parameters and class. *Medical Imaging, IEEE Transactions on*, 28(6):906–915.

Lujan, M., Paez, A., Santonja, C., Pascual, T., Fernandez, I., and Berenguer, A. (2004). Prostate cancer detection

and tumor characteristics in men with multiple biopsy sessions. *Prostate Cancer Prostatic Dis*, 7(3):238–242.

Padhani, A. R., Koh, D.-M., and Collins, D. J. (2011). Whole-body diffusion-weighted mr imaging in cancer: Current status and research directions. *Radiology*, 261(3):700–718.

Paoin, W. and Boonchai-Apisit, P. (2009). Development of surgical operation data interchange model using xml and relational database. In *Natural Language Processing, 2009. SNLP '09. Eighth International Symposium on*, pages 132–136.

Penzkofer, T. and Tempany-Afdhal, C. (2013). Prostate cancer detection and diagnosis: The role of mr and its comparison with other diagnostic modalities - a radiologist's perspective. *NMR Biomed*.

Pieper, S., Lorensen, B., Schroeder, W., and Kikinis, R. (2006). The na-mic kit: Itk, vtk, pipelines, grids and 3d slicer as an open platform for the medical image computing community. In *Biomedical Imaging: Nano to Macro, 2006. 3rd IEEE International Symposium on*, pages 698–701.

Taghva, K. and Jayakumar, K. (2009). Xml based implementation of a bibliographic database and recursive queries. In *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, pages 1073–1078.

Tuckerman, M. E., Yarne, D., Samuelson, S. O., Hughes, A. L., and Martyna, G. J. (2000). Exploiting multiple levels of parallelism in molecular dynamics based calculations via modern techniques and software paradigms on distributed memory computers. *Computer Physics Communications*, 128(1–2):333 – 376.

Türkbey, B., Bernardo, M., Merino, M. J., Wood, B. J., Pinto, P. A., and Choyke, P. L. (2012). Mri of localized prostate cancer: coming of age in the psa era. *Diagnostic and Interventional Radiology*, 18:34–45.

Vasanawala, S., Murphy, M., Alley, M., Lai, P., Keutzer, K., Pauly, J., and Lustig, M. (2011). Practical parallel imaging compressed sensing mri: Summary of two years of experience in accelerating body mri of pediatric patients. In *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pages 1039–1043.

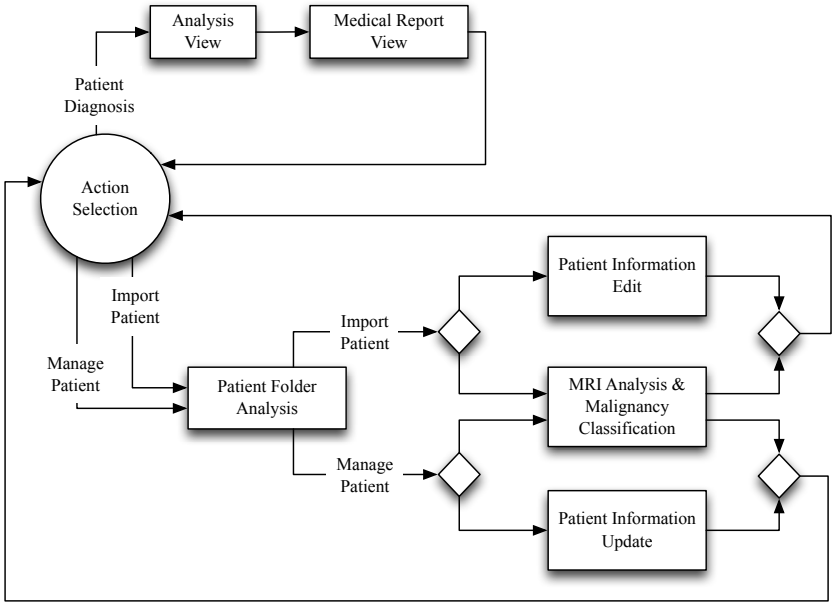

Figure 1: The Software Work-Flow in details.

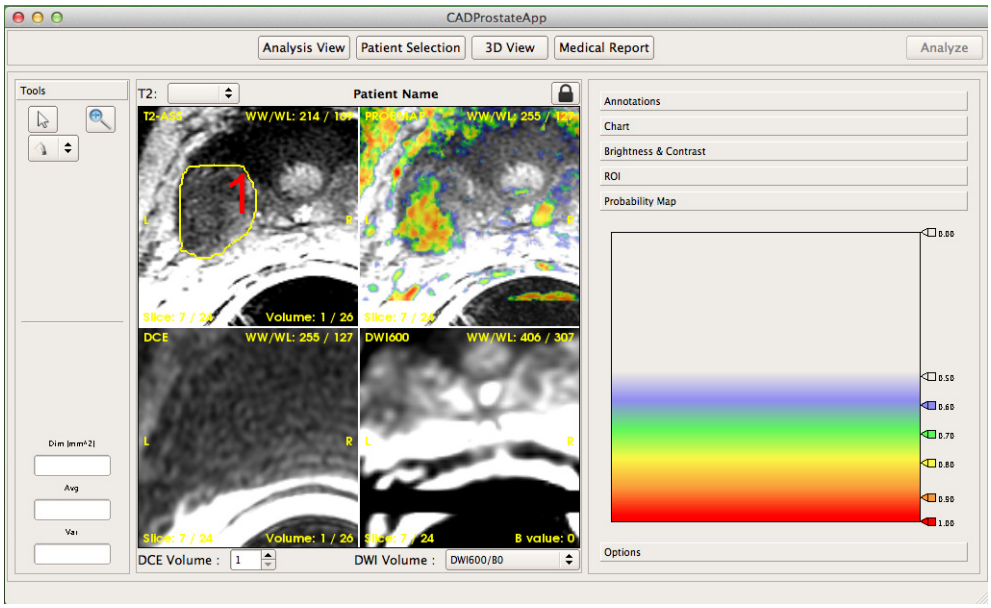

Figure 2: The prototyped GUI: the probability map can be noticed on the first quadrant of the 4 images view, and the related colors are on the right sidebar.

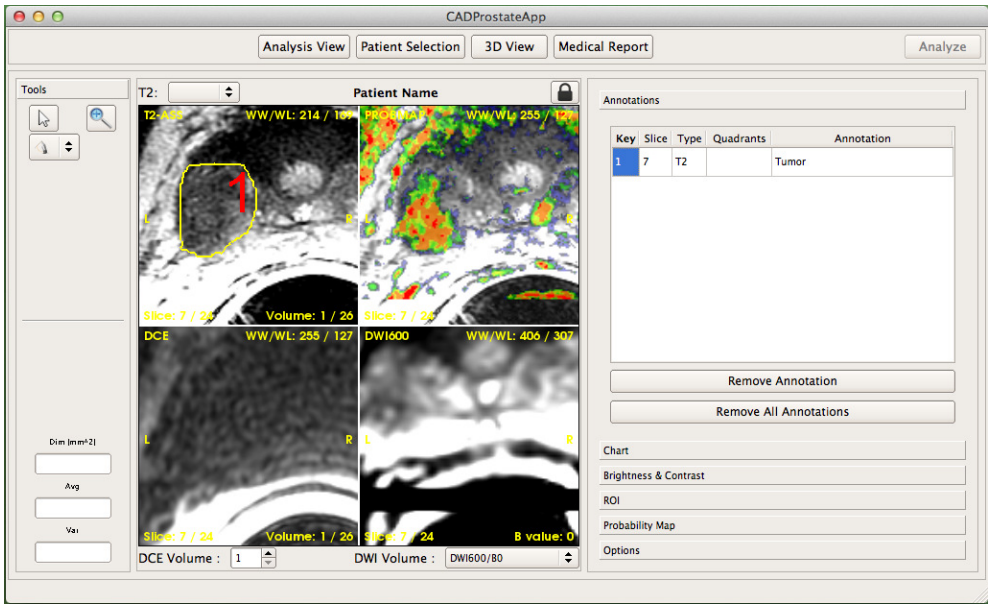

Figure 3: The prototyped GUI: annotations form is on the right sidebar.