Medical image reconstruction using the .NET Framework

Since the introduction of Computed Tomography (CT) as a medical diagnostic imaging procedure 40 years ago, there are still unresolved challenges regarding radiation dose, image quality and image reconstruction. One of the major concerns is radiation dose applied to the patient and - not frequently mentioned - the medical staff. For that reason, several new inventions have been made that improve the equipment of the CT-scanning device, for example X-ray sources and detectors systems. Recently, new developments in the computing power of modern computers have led to new approaches that enable the replacement of traditional reconstruction methods (filtered backprojection) by using iterative or AI-based methods as well as cloud-based approaches that allow both, the reduction of radiation dose and a tremendous improvement in image quality.

However, evaluating new methods require a solid development environment to allow for and simple and convenient implementation of new ideas and algorithms. Although there are several well-known and established platforms that are a good starting point for plain prototyping, an efficient execution of code inside these platforms is often challenging with regard to the management computing power and memory resources. Additionally, shipping prototypes to customers for evaluation is often not easy.

This presentation starts with a basic introduction into medical image acquisition and reconstruction, followed by a short explanation of the storage of image data using the DICOM (digital imaging and communications and medicine) standard. Afterwards it is shown how the .NET environment using C#, .NET and .NET Core, WPF and the nVidia computing platform (CUDA) can be applied as a modular prototyping platform for developers, researchers and customers in order to develop new algorithms and to conduct clinical and scientific studies.

The presentation will end with a summary of state-of-the-art development with C# and point out challenges that exist with managed runtime environments regarding memory management, performance and missing language and CLR features that would help to further improve the development within the C#/.NET environment in the light of high performance computing.