On Edge Computing for Remote Pathology Consultations and Computations

Alessio Sacco, Flavio Esposito, Guido Marchetto, Grant Kolar, and Kate Schwetye

***Abstract*— Telepathology aims to replace the pathology operations performed on-site, but current systems are limited by their prohibitive cost, or by the adopted underlying technologies. In this work, we contribute to overcoming these limitations by bringing the recent advances of edge computing to reduce latency and increase local computation abilities to the pathology ecosystem. In particular, this paper presents LiveMicro, a system whose benefit is twofold: on one hand, it enables edge computing driven digital pathology computations, such as data-driven image processing on a live capture of the microscope. On the other hand, our system allows remote pathologists to diagnosis in collaboration in a single virtual microscope session, facilitating continuous medical education and remote consultation, crucial for under-served and remote hospital or private practice. Our results show the benefits and the principles underpinning our solution, with particular emphasis on how the pathologists interact with our application. Additionally, we developed simple yet effective diagnosis-aided algorithms to demonstrate the practicality of our approach.**

***Index Terms*— Telepathology, Edge Computing, Collaborative pathology.**

I. Introduction

TELEPATHOLOGY is the practice of digitizing histological or macroscopic tissue images based on a glass slide for transmission along telecommunication pathways for diagnosis, consultation, or medical education. Pathologists seek second opinions from local experts either intraoperatively (rapid frozen section assessment of margins during tumor excision, for example) or during routine sign-out of difficult or unusual entities; in most settings, this involves the transport of physical glass slides. Hence, every pathologist should ideally be able to consult an expert via telepathology, either intraoperatively or for routine sign-out.

Today, however, telepathology is often grouped in expensive software packages that most local hospitals cannot afford, and it is practically unused for the applications that would need it the most: fast and reliable consultations, as well as multi-student, live teaching sessions; pathology is nowadays mostly taught via offline methods or via one-to-one mentor-student specimen analysis.

Our work aims at enhancing the currently booming field of Digital Pathology (DP), which can be seen as an attempt to adopt digital technologies to reduce the time and improve the accuracy of a diagnosis, for example by reducing the number of physical slide requests [1], [2]. However, developing technologies to efficiently digitalize image samples and building solutions that pathologists can easily access is a challenge.

Pathologists often analyze histological samples on a glass slide while the patient is still in the operating room. Without a telepathology solution, pathologists usually ask for second opinions from nearby experts (if available) by physically carrying glass slides, in order to minimize the time to diagnosis and report to the surgeon. When not enough local experts are available, the presence of a telepathology system could be a critical factor in providing the best care for the patient. Moreover, a telepathology system can be used to transmit high-resolution images of specimens to connected experts in order to speed up the diagnosis of more routine cases in more rural treatment locations. Finally, it might also mitigate discordance between pathologists [3]. In summary, telepathology has the potential to have a positive impact on the delivery of expert care patients regardless of the location of their surgical treatments.

We developed a new edge computing-based telepathology system, LiveMicro, that enables real-time remote control of the microscope and live histological image processing. In particular, this paper extends our previous work [4] providing more results on histological sample processing, an in-depth evaluation of the system using the real microscope, and a description of new features. Our solution enables remote collaboration and digital image sharing in addition to remote computation provided by edge computing.

LiveMicro eliminates the need to transfer physical slides and reduces to zero the travel time for remote specialists for consults or tumor board reviews. As demonstrated, this saves valuable time and expense, providing faster diagnoses resulting in better patient care.

Our system allows remote control of panning, zooming, and focus of a microscope, on *software-defined glass slides*. This is fundamentally different from a video conference, *e.g.*, for a remote surgery application; in a telepathology system, the resolution, responsiveness, and size of the histological images to transfer and pre-process are very large: both very low delays and high bandwidth are required on a dedicated virtual path that must not compete with other Internet traffic or internal

Alessio Sacco and Guido Marchetto are with the Control and Computer Engineering of Politecnico di Torino, Torino, 10129, Italy (e-mail: alessio_sacco@polito.it; guido.marchetto@polito.it). A. Sacco worked on this project as a visiting scholar in the Department of Computer Science at Saint Louis University.

Flavio Esposito is now with the Department of Computer Science, Saint Louis University, Saint Louis, MO 63103 USA (e-mail: flavio.esposito@slu.edu).

Grant Kolar and Kate Schwetye are with the Pathology Department of School Of Medicine, Saint Louis University, Saint Louis, MO 63104 USA (e-mail: kate.schwetye@health.slu.edu; grant.kolar@health.slu.edu).

campus network data flows.

We validate several simple remote computation algorithms, showing how in critical and urgent scenarios, simple computations are very effective. Our goal was to demonstrate that when there is no time to perform complex computational operations, and there are no adequate datasets to train a neural network or a deep learning system, a telepathology session allowing remote computations and remote consultations can be very effective.

Our approach does not exclude, however, the application of Neural Networks models, already largely used in many fields of pathology [5]–[7].

The rest of the paper is organized as follows. In Section II we discuss related telepathology solutions. Section IV details our *LiveMicro* system as well as all technologies used for the deployment of the architecture. Section V describes the image processing algorithms applied over histological samples, used to assist the pathologist during a telepathology session. In Section VI we describe the results obtained by measuring performance in the edge infrastructure and evaluate the quality of histological processing. Finally, we conclude the work in Section VII.

II. Related Work

Current clinical applications of telepathology include intraoperative frozen sections, routine surgical pathology sign-out, second opinions, and subspecialty consultations [8]. The term *"telepathology"* was used for the first time in a 1986 editorial in *Human Pathology*. Many studies explained the benefits brought by telepathology and limitations as well, and the majority of the analysis is still valid [3], [8], [9].

The major benefits of the introduction of telepathology that have been identified are the absence of *intraoperative consultations (IOC)* service in hospitals with no on-site pathologists and the prevention of two-stage surgeries and patient transfers. Furthermore, the retention and recruitment of surgeons in remote hospitals were facilitated because of telepathology was instrumental in their treatment decisions. Lastly, professional isolation among pathologists working alone is reduced.

Table I describes the evolution of the Telepathology class solutions, in particular, the features added and the differences among them, in accordance with the Weinstein classification [8]. The different classes are proposed sequentially, in an attempt to improve the previous systems, allowing more actions or enhancing the already present ones. First and Second classes constitute the first-generation telepathology, relatively simple systems whose imaging can be done in 2 ways: real-time imaging mode (i.e., dynamic) for class 1 [10], [11], and store-and-forward mode (i.e., static) for class 2 [12]. In both options, the histopathology field selection can take place in a remote site, which enables the remote control feature, also known as robotic. Third class systems evolved by fusion of 2 previous systems, leading to the new term "hybrid systems" [13], [14]. In the fourth class, the "virtual slide" is obtained by programming a microscope stage to automatically scan an entire slide and capture images of all microscopic fields without operator intervention. In some versions, the telepathologist can remotely control the order in which the composite images are acquired [15], [16]. Class 5 is based on an entirely different design concept and is the first telepathology system to incorporate a multisensor design [17], [18]. This technology produces miniaturized microscopes that are assembled into miniaturized microscope arrays (MMAs). This class of systems was designed to capture in less than 1 minute an entire virtual slide. In our solution, we added edge computing to improve the image processing performance, allowing us to apply image analysis before sending the result to pathologists. In addition to image operations, edge computing can be exploited for network processing.

TABLE I: Summary of Features in existing telepathology solutions: Ours added edge computing for remote consultation and remote computation.

Telepathology Solutions	Features					
	Remote Control	Real Time	Hybrid System	Virtual Slide	Rapid Processor	Edge Computing
First Class [10], [11]	✓	✓				
Second Class [12]	✓					
Third Class [13], [14]	✓	✓	✓			
Fourth Class [15], [16]	✓	✓	✓	✓		
Fifth Class [17], [18]	✓	✓	✓	✓	✓	
LiveMicro (Our Solution)	✓	✓	✓	✓	✓	✓

It is possible to find cheaper solutions as well, e.g., a system that consists of a Raspberry Pi 3 hosting a web server, Apache, that contains microscopic medical images that were captured using smartphones [19]. In another solution, the Raspberry Pi was combined with an external webcam and used to stream microscopic images to a pathologist at a remote location. They tested 3 low-cost telecytology systems: a Raspberry Pi with a webcam, an iPhone 4S with FaceTime, and an iPhone 4S with a live streaming app. Inexpensive telecytology systems are able to stream live video feeds of cytology slides from a microscope to a remote location at usable resolutions [20].

Weinberg and Randolph deployed a solution applicable in telepathology as well [21]. This project aims to enrich high-school biology education via real-time video, and the main focus is education and the interaction between high school teachers, high school students, university faculty, and students. The entire project relies on the GENI [22] infrastructure to create a high-speed Internet connection between the University of Southern California and Chattanooga, Tennessee. No edge computing is present in this solution.

The problem of transmission of large size of data, such as video, arises in other fields outside of telepathology and constitutes an active area of research. We can cite, for example, some commercial products, like *GigE* vision technology for the transmission of a video over Gigabit Ethernet [23]. The standard defines the process by which a host machine can discover and acquire images from one or more high-performance cameras to be transmitted at high speed. Although this solution provides excellent results in the video transmission, it is relevant for the one-way transfer only and lacks in supporting the remote control of devices.

Recent studies optimize the offered services, such as [24], [25], where the authors present solutions to provide a telepathology system or collaborative image viewer. However, none of them is able to guarantee a microscope remote control with high bandwidth and low delay, along with fast image processing. In fact, all these presented solutions are cloud-

based, that is optimal for storage and intensive processing, but it yields high and intolerable latency when systems need a feedback from the user, as in interactive systems.

Contrary to other solutions described, our system allows remote control of the microscope, where edge computing is the enabling technology, and integrates a collaborative image analyzer empowered with some image processing algorithms. Our solution supports high-speed data transfer with low-latencies, but aims to provide services for real-time consultations in both education and surgery scenarios using edge computing. With low-latencies, we are able to provide consultations for intra-operative frozen section requests (most commonly requested in oncological surgeries). By low-latency system we mean a program able to reduce latency to the order of hundreds of milliseconds, considered as a threshold for enabling the user interaction [26].

III. Use Cases: Teaching, Diagnosis and More

Remote real-time consultations on rare or newly recognized diseases. Patient care relies on rapid and accurate diagnosis in the pathology laboratory. In certain critical situations, the time to diagnosis can literally be life-saving, including timely recognition of bacterial organisms in cerebrospinal fluid (meningitis). Often, these diagnoses are made by a pathologist using a microscope to analyze biological material (cells or tissue) on a glass slide, which is subject to human error. The accuracy of the diagnosis depends on the experience of the on-call pathologist, who may cover a broad range of subspecialties. Another common scenario occurs during surgery when a quick and accurate pathology assessment (frozen section [27], touch or smear cytologic preparation) necessarily defines the next treatment steps by the surgical team. In all these scenarios, LiveMicro would offer a quality-oriented platform for remote consultation by connecting more experts to cooperate in the pursuit of the best possible diagnose.

Continuous medical education. Pathology today is taught to medical students, residents or in a session of continuous medical education in a one-to-one form with microscopes with double or triple oculars. With LiveMicro, multiple students (and a teacher) could access the same live telepathology session in an online classroom environment. Each connected user could manipulate individually a virtual instance of the microscope. Using our software, multiple connected users can perform actions on a captured image, *e.g.*, zoom, pan, or save the images to their local device.

Remote (complex and resource intensive) computations. In addition to multi-session (live) teaching and rapid remote case consultation for pathology, we foresee the success of LiveMicro in furthering the research and educational goals. Research microscopy imaging data takes many forms, ranging from static large virtual slide images, often collected in single axial planes, to more complex and resource intensive datasets including multiple axial planes (Z-stacks), multiple data channels (multi-color fluorescence imaging), and time-course data involving both multiple image planes and fluorescence channels. Storing these images often require several gigabytes. Consequently, very high-definition image

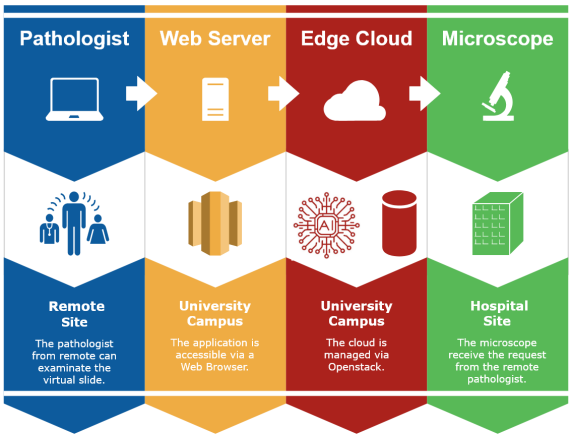


Fig. 1: Remote pathologist can access the microscope through the web application, that leverages Edge Cloud computation.

transfer for analysis, collaboration, or consultation requires significant resources and time using conventional network channels, *e.g.*, the Internet. Furthermore, collaboration in the research microscope laboratory is increasing as multi-PI research programs are becoming essential in the biological sciences. Such activities depend on data communication not only for consultation among colleagues, but also for the use of common computing resources often housed in individual laboratories or core facilities.

With the increased capabilities offered by the edge computing-based LiveMicro, researchers and (bio)medical students can now use their data captured and store them in a "virtual slide", with confocal and super-resolution. This is relevant for a series of applications, for example, scanning electron microscopes for complex cell tracking, automated (observer-independent) morphometric analysis, co-localization, 3D reconstruction, and image deconvolution analysis.

IV. LiveMicro: Edge-Based Architecture

In this section we describe the LiveMicro system and its design and implementation details. Our design has focused on allowing (i) *remote computations* and (ii) *remote consultations*. In spite of pathology applications, we argue that any field that uses a microscope, for example microbiology, may benefit from our system. Remote consultation entails the possibility for pathologists to access through our web interface to a live session of a microscope and remotely control its firmware and its functionalities. Typically, pathologists ask for zooming, panning, focusing, or taking snapshots of histological samples. The entire application is accessible via a web browser (Figure 1).

A. Front-End and Plugin Design

Our design goal is to allow pathologists to access the microscope in a user-friendly manner, merely through a web browser. For this reason, as shown in Figure 1, the entry point for the entire system is a web server, which acts as a portal through which users connect to the ecosystem and join, start, or terminate one or multiple telepathology sessions. The choice regarding the front-end design goal was to create a component as lightweight as possible, keeping a user-friendly and intuitive interface. The web front-end is implemented in AngularJS,

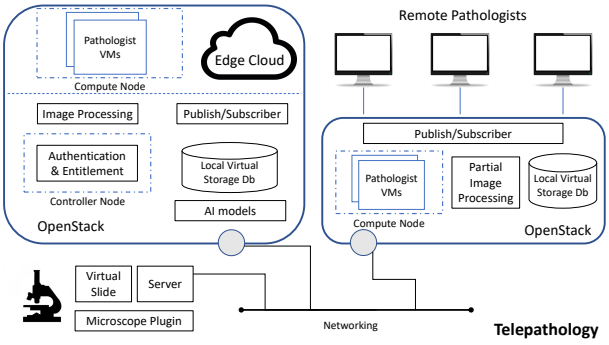


Fig. 2: Services are spread across the infrastructure, the microscope uses a dedicated machine and a dedicated hardware for capturing samples.

and the slide visualization is manged by the OpenSeadragon JavaScript library [28].

At the other end of the telepathology session, a computer runs a modified version of *μManager* — often named Micro-Manager, as in microscope-manager — an open-source package for configuring and controlling a fairly large amount of commonly used microscopes. We can state, for example, Leica, Nikon, Olympus among the microscopes supported, and Hamamatsu, Canon, and Nikon for the cameras. However, although Micro-Manager provides this extensive support in the standard version, it is also possible to write new device adapters (i.e., "drivers") for the new hardware, or even improve existing ones.

Originally *μManager* did not support network connectivity nor edge computing functionalities. Our modified instance of *μManager* can be plugged to a physical machine attached to a microscope, to handle data marshaling between the network and the microscope firmware. In our prototype, we use an Olympus IX81 [29], one of the microscopes whose interface is compatible with Micro-Manager. LiveMicro can also operate on an emulated version of the microscope, that may result a very effective tool to scale, for instance for pathology medical education.

Since pathologists (in training or at work) need to examine tissues among diverse microscope lenses, a simple image snapshot is often not enough. Live streaming of the sample under consideration is hence necessary to have immediate feedback and make the system usable. We use *ffmpeg* [30] to encode and transmit videos from our Micro-Manager Plugin (from now on denoted as, Plugin) to the LiveMicro Server, while on the web-page, our *WebRTC* [31] interface is responsible for receiving and showing the video. We provide further details on the live streaming process in Section IV-C. Our Edge Cloud infrastructure pre-processes the stream compressing its payload. The compression is not performed on the Plugin, but in a second phase, thereby videos can be stored and retrieved at a later stage.

B. Core and Edge Cloud Management

To control large pools of storage, compute, and network resources between the web server and the Plugin, we deployed our own Edge Cloud infrastructure (Figure 2), modifying OpenStack [32], a well-known open-source Cloud Computing platform. We associate each user of a telepathology session to a VM that provides network and node functionalities, *e.g.*, CPU-intensive algorithms on the histological imagery.

The architecture and the object model that we propose will leverage the computing paradigm known as Edge (or Fog) Computing [33]. In an Edge Computing system, much of the processing takes place in a process at the edge of the network (as opposed to the core of the network as in the Cloud Computing paradigm). It is inefficient to transmit all telepathology generated data to the cloud for processing, data query or analysis; doing so requires a great deal of bandwidth and all the back-and-forth communications between the sensors and the cloud would adversely impact performance, rending the application unusable. Our proposed infrastructure will enable flexibility in the mechanisms that regulate the adaptation (creation, monitoring, and migration) of virtual network resources carrying glass-slide imagery.

Our architecture at the edge demands at least two nodes in charge of launching the core management features: the *controller node* and the *compute node*. The controller manages the resources available in the infrastructure, and the compute node is where the VMs and their bookkeeping are installed; the networking service agent then connects all telepathology instances to their isolated virtual networks providing besides security services to instances.

The best hosting machine is chosen according to some configurable criteria, *e.g.*, the usage of machines at that moment or hosted application requirements. OpenStack selects by default the hosting machine regardless of the application logic. But in our scenario, we forced the controller node to choose a node close to the requested microscope, in order to guarantee a low delay. Specifically, we adjusted the default cloud orchestration mechanisms to better support edge computing applications, modifying the VM scheduler to handle multiple telepathology sessions.

The core process is the LiveMicro Server, where most of the telepathology application logic resides. It receives requests from either the web server or the microscope plugin, and it is responsible for determining the following operations: it manages live sessions, it communicates with the database and decides when it is time to create or destroy a VM, according to the business logic. In our system, to reduce the startup delay when a VM is created, we keep a pool of VMs already deployed. In such a way, when a user needs to operate on a VM, the system can exploit the existing VMs instead of creating new ones, hence reducing the response time. Moreover, the LiveMicro Server provides the services needed to control the microscope remotely. This program holds also the management of the prior (live) sessions and implements the Node.js REST API called by the emulator plugin and the desktop application. We select the Node.JS environment to handle the requests as it connects the ease of a script language as JavaScript with scalability, fast implementation, and high speed of the code execution [34].

Each VM hosts a telepathology session client, which acts as a proxy: it receives requests from the LiveMicro Server through gRPC protocol [35], elaborates them, and in case sends them to the Plugin. If the response is already in the

VM, *e.g.*, the client is working on an already snapped image, the request is not forwarded to the Plugin. This cache layer is fundamental to avoid overloading the real microscope.

The image and video processing of each telepathology session occur across multiple VMs. Hence, each client can ask different processing on the same virtualized histological image, at the same time. The system assures that actions of a user do not affect the analysis of another pathologist working on the same sample, as the operations take place in different VMs. The doctor is free to work independently as long as the action does not modify the microscope settings. In this latter case, the actions are enqueued and synchronized, and all the clients of the sessions will be affected. This strategy is based on the intuition that users are more likely to require image and video processing rather than microscope configuration. The synchronization is necessary to prevent the access to the shared resource (the microscope) from multiple agents, that can potentially lead to an inconsistent state. However, as demonstrated in our evaluation (Section VI), the time to execute actions is significantly smaller than transmission and processing time, proving that, when ten users are cooperating, the waiting time is negligible.

Image Management and Visualization. The pathologist can thus analyze images that are captured in real-time or that were previously saved. In the former case, there is not enough time to generate a WSI and, for this reason, the system sends only the slide as captured by the camera. If requested, the image is processed, but the algorithm is limited to the viewed image only. Further details on the streaming of the slides are provided in the following Section IV-C.

In the latter case, i.e., offline digital slides, the web application uses the OpenSeadragon image viewer [28] for navigating and zooming over the slide. On the server side, specifically on the VM, our program returns tiles of specified coordinates and zoom levels on demand. Albeit as mentioned earlier these images are very huge and require enough disk space to be stored, we optimize the RAM usage by loading not the entire image, but only a small portion of it, i.e., a tile. These tiles are generated on demand using a PyramidIo-based tool [36] that generates them. PyramidIo allows reading WSIs of different formats and facilitates the rapid retrieval of subregions of the image. The tiles could also be pre-generated and saved on the file-system for future usage. Nonetheless, we experienced that performance difference between dynamically-tiled and statically-tiled viewing sessions is imperceptible for the user. Moreover, the use of dynamic tiling avoids an additional step in the workflow and eliminates the storage overhead of storing tiles.

Alongside the management of smaller tiles, the VM provides a cache layer between the client and the storage, so that multiple requests for the same area are immediately replied. In such a way, the time to process a request is shortened. Additionally, the web browser, through the OpenSeadragon library, is a supplemental layer for caching the image bytes, in order to reduce the response time.

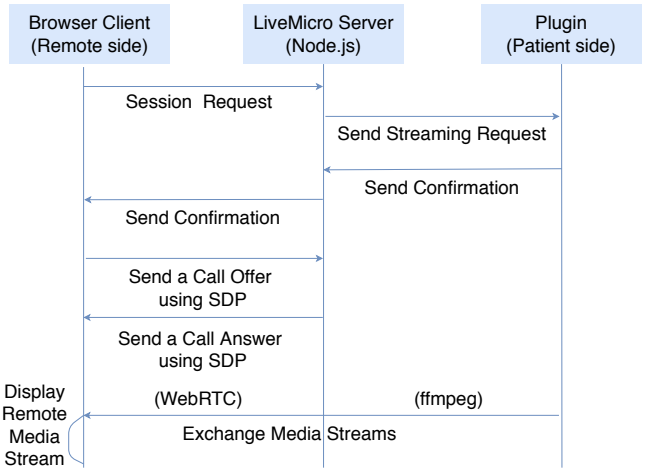


Fig. 3: Signaling and Interaction of Video Streaming.

C. Live Streaming Architecture for Immediate Medical Feedback

Obtaining immediate feedback from the system is extremely important in these solutions, and live streaming from the microscope camera is an effective way to give this feedback to the medical doctor. In our system, indeed, the remote pathologist can manipulate the microscope in search of the best possible sample for the diagnose. In order to provide the video streaming to all the pathologists connected to the live session, we leverage *ffmpeg* as an encoder, and *WebRTC* as a de-multiplexer and for showing the video in the web-page.

Frames acquisition and transmission. At the plugin side, the frame encoding occurs via *ffmpeg*, which reads from files the frames captured by Micro-Manager and sends them as a stream to the LiveMicro Server. This tool allows specifying significant video options, such as aspect ratio, video codec, frame rate. These parameters are set to guarantee low delays and to achieve zero losses in the capture, therefore moderately sacrificing the video resolution. Nonetheless, as in the edge computing paradigm, the LiveMicro Server is in the proximity of the Plugin, which allows not to drastically downgrade the video resolution in favor of the latency. Besides, the stream is directly sent to the LiveMicro Server rather than passing through the user's VM, in order to further reduce the application delay. It is also worth noticing that very high-resolution images, when needed, can be obtained by means of offline downloads.

On the LiveMicro Server resides *WebRTC*, which manages the transmission of the video stream to the final user. It is responsible for communicating with the web browser in order to agree on the best bitrate that adapts to the network conditions. This service receives the stream of data from the Plugin and forwards it to clients interested in receiving the video. In case the client asks for a reduced bitrate compared to the *ffmpeg* stream, *WebRTC* handles this difference and stores the data temporarily in order to synchronize the incoming stream with the outgoing one. Moreover, it receives the commands from the user and forwards them to the microscope, dispatching the requests in the same order they are received. *WebRTC* comes up with several components for different functionalities provided. Among them, we can enumerate signal processing,

codec handling, security, bandwidth management, to name a few. The services such as STUNA and TURN, provided by *WebRTC*, allow the communication to hold even in the presence of firewalls and NATs, very common in real scenarios.

Figure 3 summarizes the messages we defined to instantiate the streaming from the microscope to the connected user, as well as signaling requirements and interactions. *WebRTC*, with its module PeerConnection API, is responsible for managing the full life cycle of each client-microscope connection at the remote site, by encapsulating all the connection setup, management, and state within a single interface. When the user joins a live session, it contacts the signal server (LiveMicro Server) to create the session and a secure channel. Once the server receives the request, it contacts the Plugin asking to start the streaming session. At this point, the Plugin initializes the samples capture procedure and sends the confirmation back, which is eventually forwarded from the LiveMicro Server to the client. Then, the client sends an *offer* message containing a description of the streaming session according to the Session Description Protocol (SDP) format. This description includes the streaming communication parameters, such as the types of media to be exchanged, codecs and their settings, and bandwidth information. Upon receiving the offer, the LiveMicro Server creates an answer for connecting to the endpoint. After having checked the information in response are correct, the peers establish a connection, and the Plugin can finally send the media stream to the client, passing through the server. The first part of the communication, i.e., between the Plugin and the server, occurs via *ffmpeg*, while the second part, i.e., between the client and the server, is managed by *WebRTC*.

V. Image Processing

Typical image processing operations performed on the samples can include color deconvolution, nuclear and mitotic counts on tumor cells, tumor pattern detection as the user moves the image, *etc.* . Machine learning algorithms are under rapid development to assist with the latter. As other software that allows such image processing, our application is written in Java and uses standard libraries such as ImageJ and OpenCV.

Antibody markers such as Ki-67 are used to measure the rate of tumor proliferation by immunohistochemical methods, particularly in neuroendocrine tumors. Currently, quantification is done predominately by hand-counting positive and negative nuclei on selected regions of the tissue. While software packages that distinguish immunohistochemically-positive and negative cells exist, they do so on merely single photographs of selected regions. Our solution allows the entire virtual slide to be processed (more than a single photograph) using high computational power present within a regional network, *i.e.*, at the edge, or remotely, using virtual network paths with reserved bandwidth.

Figure 4 shows an example of image processing aiming to count the affected nuclei. The counting is recomputed every time the image is moved or zoomed. The knowledge of the percentage of cells activated by different markers (represented by different colors), can be used by the pathologist to assess the tumor grade, a critical parameter for the course of an active surgery or a treatment.

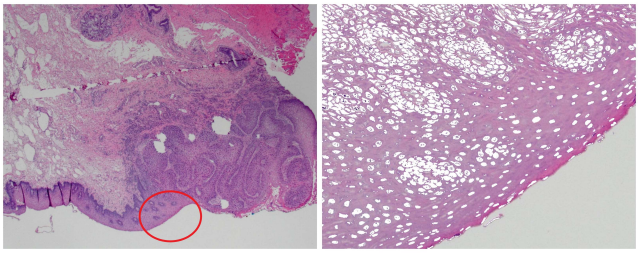
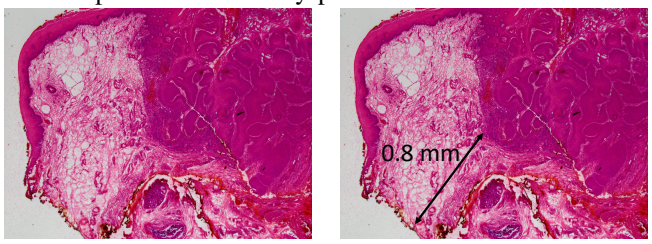


Fig. 4: (Left) View of invasive squamous cell carcinoma of a tongue, with tumor area highlighted in the red circle at 2X magnification. (Right) Image after the application of nuclei detection algorithm at 10X magnification: in addition to the image, information about percentage of marker co-expression in the sample is automatically provided to the web interface.



(*a*) (*b*)

Fig. 5: Example of distance tumor to margin operation by means of an algorithm that automatically detects the margin. Tumor is automatically detetcted as well, when the pathologist selects the area of interest. Distance is returned in “mm”.

In Figure 5, we measure the distance from the tumor and the margin, inked in yellow at the left side. The pathologist marks the tumor, and the algorithm automatically detects the margin and computes the distance between the two points. As demonstrated by previous solutions [37], this information is extremely significant, but, sometimes it is computed by hand by pathologists. The distance from tumor-to-margin is typically reported in oral cavity malignancies (usually, squamous cell carcinoma), as it has been shown to impact prognosis when the margin is less than 5 mm. Before the gross specimen is sectioned and sampled in cassettes for histology, inking is performed, making it easy to detect it for a computer-based algorithm. Automatic calculation of the closest distance between the tumor and specimen margin (edge) can greatly decrease evaluation time and reporting accuracy. The distance can also be used in breast cancer specimens, with particular interest on the distance from an invasive and in-situ tumor in the standardized reporting forms.

A. Image Processing for Faster and Accurate Diagnostics

The edge computing paradigm allows us to perform fast and intensive processing otherwise impossible to obtain on a local machine. We provide the overall procedure in Algorithm 1, highlighting the different operations performed when the image is locally saved on the server or not. With the aim of reducing response time, and so the diagnosis turnaround time, we first check whether the requested image is already in memory. If so, our software retrieves it, and our processing algorithm (if configured by the user) is applied (lines 6-7). The

user can also specify if a compression needs to be applied over the final image. In this case, the image is sent after a lossless compression. In case of urgent analysis, a lossy compression algorithm can be requested and applied, where the compression quality is a customizable parameter. The compression quality is a value between 0 and 1: a compression quality setting of 0.0 means that the highest compression must be applied, while a setting of 1.0 requires images to be of the best possible quality. By default we use 0.75. In a lossy compression schema, the compression quality determines the trade-off between file size and image quality, by choosing quantization tables when writing JPEG images. For lossless instead, the compression quality is used to control the trade-off between file size and time taken to perform the compression, by optimizing row filters and setting the ZLIB compression level when writing the final PNG image.

If the image is not in memory, it must be retrieved from the database, and then we follow the same procedure mentioned above (lines 8-11). In this case, however, we locally save the image in order to avoid the transfer delay in case the pathologist will come back to work on the same sample.

The presented procedure works for offline images, while for images snapped during the live session, a lossy compression algorithm is necessarily applied (unless otherwise specified) as they have to be retrieved from the microscope (lines 12-16).

In the following, we describe two examples of image processing which can be applied either in real-time for captures of the microscope or offline for stored samples. Before the application of such algorithms, it is often required to regularize the image, typically via a stain normalization, in order to increase the accuracy of following image operations.

Algorithm 1 Image Edge Processing: Overall procedure.

1: Let *I* be the requested image by client *C*
2: Let A_c be the compression algorithm (lossy or lossless)
3: Let *L* be the level of A_c
4: Let I_c be the final image for client *C*
5: **if** I is a saved image **then** ▷ Offline image
6: **if** I is already in memory **then**
7: $I \leftarrow$ the image and apply algorithm
8: **else**
9: $I \leftarrow$ the image from the database
10: Save I in memory, and apply algorithm
11: $I_c \leftarrow$ apply A_c with level *L* and send
12: **else** ▷ Real-time image
13: $I \leftarrow$ the image from the microscope
14: Save I in the database and in memory
15: Apply algorithm
16: $I_c \leftarrow$ apply A_c with level *L* and send

Stain Normalization. The procedure of staining the tissue sections affects the appearance of H&E histology samples, and it can vary significantly across laboratories, but even across samples within the same lab. Although this variability only partially limits the interpretation of images by pathologists, it may affect the result of subsequent image analysis algorithms. To address this problem, Stain Normalization (SN) algorithms are able to match stain colors of histological images with a given template [38], [39]. Formally, SN consists in transforming an image I into another image I^* via a function f, $I^* = f(\theta, I)$, where θ denotes a set of parameters extracted from a template image t. θ includes the parameters that are generally defined to capture color information of the main stain components in the image, such as H and E. The final result I^* is a stain-normalized image whose distribution of colors resembles the ones of the template t.

In this manuscript, we considered a state-of-the-art algorithm based on the method published by Macenko et al. [38]. It is applied as a reference method in a large number of applications, such as in community challenges [40], and other studies [41]. However, since this algorithm is time-consuming and not always necessary for the methods described in the following, it is applied only when specified by the user.

Counting nuclei. In Figure 6, we describe an example of methodology we validate in this manuscript, used for the *count nuclei* task. First, the image is retrieved either in real-time from the microscope or offline from the database/ memory, and the user labels the Region of Interest (ROI). The ROI boundaries are set using our interface, which provides commands to select an area on the image. However, if not specified, the program assumes the ROI as the whole image and continues the pipeline in the same way. One of the advantages of edge processing is indeed the possibility to perform computations on the whole slide image, which would be extremely power and time consuming on local machines. After converting the image to grayscale, a Gaussian filter is applied to blur the image and to reduce noise. The Gaussian filter is a linear filter, usually adopted for low-pass frequency filters, since its effect is to remove high spatial frequency components from an image, thus reducing the contrast of the sample. In our experiments, the value of sigma in the Gaussian filter was 0, and a kernel size of 5×5 pixels was used.

As the third step, we apply segmentation through the Otsu's method to perform automatic image thresholding [42]. This algorithm finds the threshold that minimizes the weighted within-class variance and returns the image whose pixels are either black or white. On the obtained image, the detection of contours is indeed easier, due to the high contrast between black and white pixels. Thus, the contours are easily extracted using the algorithm [43], and we draw them on the final image so that pathologists can observe them even in the RGB colored image. As additional information, we provide the number of found contours n, which is fundamental for the pathologist.

If necessary, the pathologist can decide to show the nuclei centroids, for the sake of improved visibility and clarity of the image. In this case, nuclei centroids were detected as the moments of centroids of connected targets in a binary image, through the Green's formula [44].

Tumor to margin distance. Algorithm 2 formalizes the operations performed to obtain the distance from tumor-to-margin with an uncertainty of g. Once the ROI, defined by the main malignant area within the pathology image, is set, the pathologist marks the tumor as a point m in the image. Considering that the margin of the cell is usually inked, e.g., in yellow as in Figure 5, a computer algorithm can rapidly

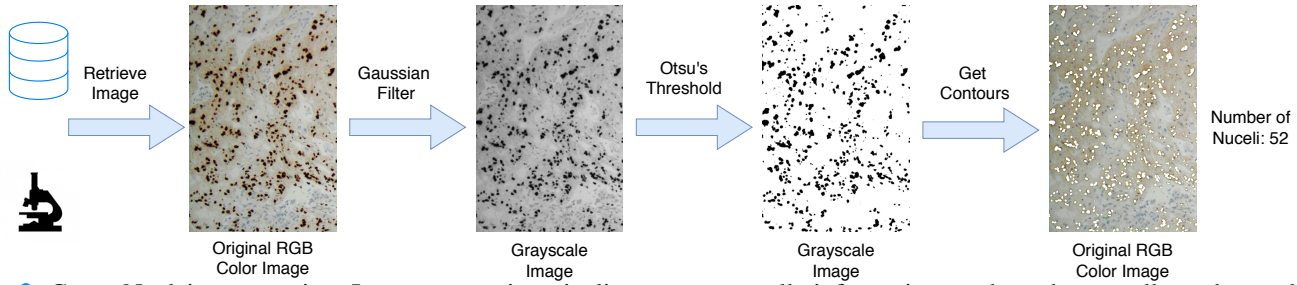


Fig. 6: Count Nuclei computation. Image processing pipeline to extract cells information, such as the overall number and the position inside the image.

Algorithm 2 Tumor-to-Margin Distance Computation.

1: Let I be the requested image by client
2: Let g be the granularity of the measurement
3: $I_1 \leftarrow$ ROI area of the image$I$
4: $m \leftarrow$ mark the tumor point
5: $CO \leftarrow$ get the contours in I_1
6: $d \leftarrow 0$, $ext \leftarrow false$, $p \leftarrow 0$
7: $v \leftarrow \min$(distance between m and the borders of the I_1)
8: $r \leftarrow v/2$
9: **while** $d = 0$ **do** ▷ We have the answer if d is not 0
10: $circ \leftarrow$ circumference with center m and radius r
11: **if** $ext \neq true$ **then**
12: **if** $circ$ intersect CO **then**
13: $r \leftarrow r - g$
14: $ext \leftarrow true$
15: **else**
16: **if** $circ$ is internal to CO **then**
17: $p \leftarrow r + g$
18: **else**
19: $v \leftarrow r - g$
20: $r \leftarrow (p + v)/2$
21: **else**
22: **if** $circ$ intersect CO **then**
23: $r \leftarrow r - g$
24: **else**
25: $d \leftarrow r + g$ ▷ Final result
26: **end while**;
27: $I_d \leftarrow$ draw distance d on I_1
28: **return** I_d and d

detect the margin contours using a procedure to identify pixels of a specific color. Then, the maximal distance initializes v, which is computed as the minimum length between m and the edges of the ROI. This value is quickly obtained through four comparisons of location m, with the four borders, and is then multiplied by the ratio to get the distance in "mm". The minimal distance p is instead initialized to 0.

After this first initialization phase, an iterative search starts, aiming to find the distance as the radius of a circumference centered in m. The first part (lines 11-20) is a binary search of a circumference that intersects the cell's margin. Binary search is a well-known approach to find the position of a target value within an array. In this case, our algorithm has the contour CO as target and compares this target to the circumference $circ$ with a radius equals to the middle of all possible distances, i.e., a number between p and v. If $circ$ and CO are not crossed, the half in which the target cannot lie is eliminated, and the search continues on the remaining half, again taking the radius r as the middle element to compare, and repeating this until the target value is found. By leveraging binary search, we can reduce the number of iterations [45], compared to other iterative search techniques that are more time-consuming.

When the circumference is found, ext is set to true, and the second part starts. Given the valid circumference identified at the end of the first section, we then try to minimize the value of r to reach the minimal distance between the target and m. In fact, the $circ$ found is clearly valid, but it might not be the smallest circumference that meets the target shape. For this reason, in lines 22-25, we iteratively decrease r until the derived $circ$ resides outside the target. At this point, we are sure that the distance is the last value of r plus the granularity g. Finally, the algorithm returns the distance with an uncertainty g, and this value is reported in the image, in a way that pathologists can use this information for subsequent operations.

As can be clearly seen, the granularity value g significantly affects the number of iterations and, consequently, the overall time. However, despite the default value of 0.1 mm that we set since it is reasonable in most of the cases, it can be customized by the pathologist to tailor the specific use case.

VI. Experimental Setup and Results

To establish the advantages brought by our design, we developed a testbed deployed both on our own servers and on CloudLab [46]. Firstly, we used the emulated version of the microscope to test our edge-based telepathology system. Secondly, we evaluate the algorithms implemented in terms of accuracy and computation time. Thirdly, system performance and resource usage are measured in order to assess the practicality of our approach. Finally, we conduct experiments over a real microscope to quantify the overhead due to physical device effects. Throughout this experimental phase, all services are installed on VMs deployed by the OpenStack orchestrator across three physical machines.

In a typical scenario, the pathologist requests to join a session and remotely controls the microscope. To test our system performance, and to cope with the lack of a real end-user in the emulated version, we replaced the front-end web server with a request generator. The messages are sent directly to the LiveMicro server, which multiplexes and demultiplexes network and application requests to/from the proper VM. In this context, the program calculates the encountered end-to-end latency within our edge cloud when responses are received.

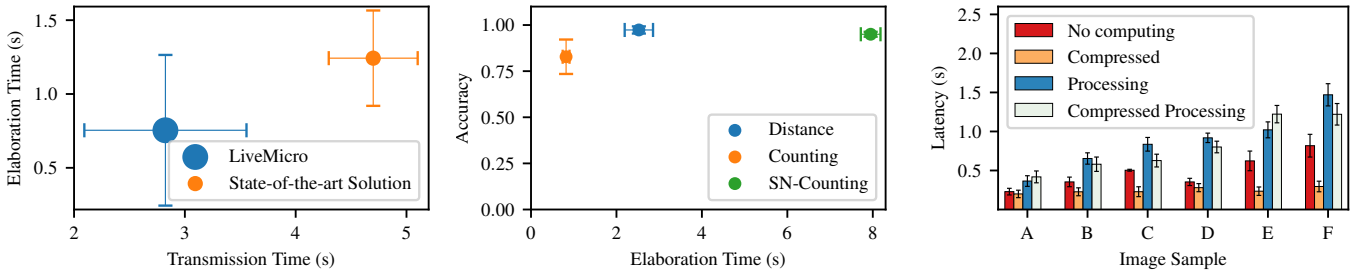


Fig. 7: Edge Computing Advantages: (a) Elaboration and Transmission time of image processing performed by LiveMicro and a representative legacy solution. (b) Trade-off between accuracy and time for distance tumor-from-margin algorithm and tumor detection counting. For the latter percentage is considered output if resides in a 5% range around the true value. (c) Time necessary for different operations to be performed at the edge plus image transmission time for samples with varying size.

To evaluate our system, and in particular the computer-aided image processing, we rely on real microscopic samples retrieved from an online database [47]. On the other hand, in the microscope emulated version, standard images from the default camera of Micro-Manager are used. Despite not being representative for assessing the validity of the processing, the latter setting is especially effective to measure the latency and scalability of the system when microscope settings are not involved.

A. System Performance Analysis

One of the main benefits of adopting edge computing in telepathology is the use of computational capacity, not available on the microscope, to pre-process histological images and help the pathologist team with their diagnosis, as well as to speed up their image transfer. To validate such advantages, we process a variety of images and quantify the expected system performance improvement. On the local testbed, we run our microscope emulator, while the servers are hosted on Cloudlab bare-metal machines.

We begin by quantifying the time involved for a pathologist to receive, compress, and process an image. In this case, the image processing entails the nuclei count, necessary to assess whether or not it is a tumor and the tumor grade. Tumor cells are labeled with different histological markers or antibodies, such as Ki-67. We consider two use cases (Figure 7a): edge image processing (LiveMicro) and local image processing (State-of-the-art Solution). In the *LiveMicro* use case, the image elaboration is performed at the edge of the network, by a machine in local proximity of the microscope. In this setup, the Plugin and the edge servers are connected to the same Local Area Network (LAN) in the Saint Louis University campus. Instead, the client is in the same city but not on the same LAN, and connects to the servers through the telco infrastructure. After the first processing, the result is then compressed and sent back for use. In this context, the elaboration occurs on host machines running Ubuntu, Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz, while the VMs are limited to 1VCPU, 2GB RAM and 20GB Disk. Conversely, in the case of a *State-of-the-art Solution*, the client receives the original image, then it runs the image processing algorithm on it. In this scenario, the image sent needs to be uncompressed, since the calculation is better performed on the original version, where the pixel information is maintained as close to the original as possible. On the contrary, an elaboration on a compressed image can lead to erroneous diagnosis. The testbed is similar to the previous case, the client is in the same city but connected through a different LAN. Host machines are Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz. As evident from Figure Figure 7a, the proximity of processing offered by edge computing enables a lower latency, hence better addressing the concerns of the response time requirements. Both elaboration and transmission time are greatly improved compared to local processing, due to a shorter transmission path between the microscope and the edge machines, which are also more powerful than conventional desktop PCs. We can also argue that in a more traditional cloud-based solution, servers could provide higher computation capabilities and hence handle complex workloads, such as image processing, in a relatively short time. In such a way, the images would be first elaborated by a server in the cloud and then sent compressed to the client. However, in this case, the latency between the microscope and the server would have a more considerable impact leading to higher transmission time.

We also measured the quality of the algorithms implemented for assisting the pathologist during the diagnosis. Figure 7b reports the accuracy and computation time of the algorithms that were previously exposed in Section V, where *SN-Counting* denotes the operation of counting nuclei preceded by a stain normalization (SN) process. The accuracy metric is the ratio of the number of correct estimations to the total number of input samples. In the case of the counting nuclei process, we define as correct a value that resides in a 5% range around the true value, while the distance must differ for less than 1 mm to be considered correct. The algorithms are tested over 100 samples and results are averaged over 50 runs. Images for the distance algorithm are obtained from the online and publicly available database [47]; given the metadata included, the ground truth is computed as a distance of two points in the pixel space, then multiplied by the image scale to obtain the final measurement in “mm”. On the other hand, for the counting nuclei, we utilize the Kaggle dataset [48], which also provides the solution so that the ground truth and the accuracy can be computed straightforwardly. The two datasets are then reviewed by our team of pathologists.

We can conclude that distance tumor-to-margin is a more effective method, and in fact, the accuracy is higher compared

to a tumor detection algorithm. The problem of detection nuclei methods resides in the variety of colors of samples and inks used to highlight tumors, leading to a solution that hardly fits all the cases. In fact, when SN occurs ahead of the counting, the accuracy of the process rises to 97%. However, this benefit comes at the cost of higher processing time. For this reason, we choose to perform the stain normalization only when required and offline, as it would be too cumbersome for real-time operations. The impossibility of achieving 100% accuracy partially resides in approximations that are inherent to the methods and partially overlapped cells. Although other implementations of stain normalization are available, such as in [39], [49], our main goal is to demonstrate the applicability of such method even in the context of edge-computing. LiveMicro can be extended with other algorithms if provided by the user. On the other hand, borders are more easily detected by computers, and the distance may be approximated more precisely. In fact, the response is extremely accurate, since the result differs less than 5% from the true value in 98% of the trials. Furthermore, we can observe that time to obtain the distance is slightly higher than time to compute the count of nuclei. We re-conduct this behavior to the intrinsic complexity of the distance computation algorithm, in which the refinement phase is notably time-consuming.

Further, Figure 7c evaluates LiveMicro in different circumstances by reporting the latency to process image samples of various sizes. Image sizes range between 38 kB (image sample A) and 14 MB (image sample F). Latency values are the average between cases in which the image is retrieved from the database (worst-case scenario), and the image is cached on the server (typical scenario). Four situations are taken into account: (i) *No computing*, no processing on the sample is performed, (ii) *Compressed*, the image is sent (lossless) compressed but no processing on the sample is performed, (iii) *Processing*, one image elaboration algorithm (counting nuclei) is applied on the sample sent without compression, (iv) *Compressed processing*, one image elaboration algorithm (counting nuclei) is applied on the sample sent after compression. Results confirm that, in the latter case, the system can benefit from the high computation power of servers and the low latency due to compression. Clearly, compression diminishes transmission time because of the smaller compressed payload. Nevertheless, these results demonstrated the significant benefits of even a simple pre-processing performed at the edge to handle a telepathology session.

Furthermore, we analyze the system's degradation performance when more clients are working simultaneously. Figure 8d shows the average waiting time, i.e., the time elapsed since the request arrives at the Plugin and its execution, for four kinds of actions performed on the microscope. We can observe that the time slightly increases when the number of users increases, but even the time needed for moving the z-stack (the more time-consuming action) is negligible compared to elaboration and transmission time. This result ensures that multiple users can cooperate smoothly and that the bottleneck resides in the processing and transmission of slides.

Lastly, to prove the usability and the correctness of the algorithms applied, the system has been validated by a team of ten pathologists. The response was extremely positive, especially in terms of ease of use and reactiveness, and in fact, the compression after processing resulted as a good way to shorten latency. In this evaluation, the pathologists try to access the same microscope in a close area, attempting to control it and to perform operations on the image. The latency obtained with LiveMicro enables interactivity and usability, confirming the numerical results also from a user perspective.

B. LiveMicro performance: resource usage analysis

We envision LiveMicro to run in hospitals with (some but not severe) budget constraints, which often lack expensive hardware but are provisioned with a standard medium size datacenter. To this end, we run a set of experiments to assess the base performance needed to run LiveMicro smoothly. We also present some results on resource load related to create and maintain a LiveMicro session on a VM.

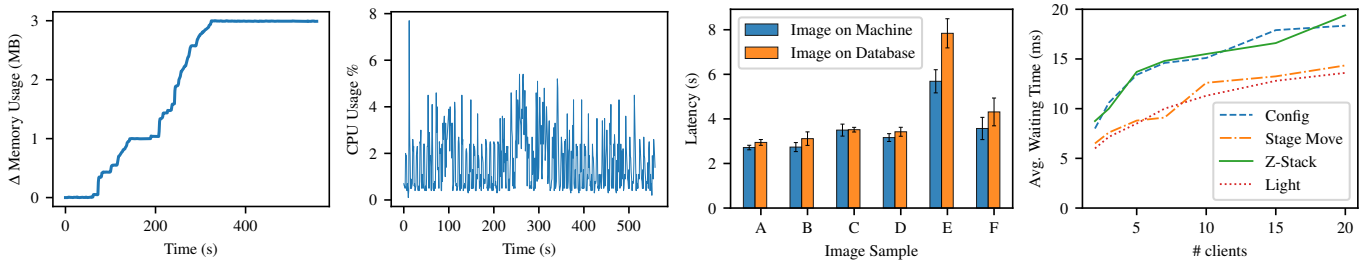
As reported in Table II, the overhead of bootstrapping a new LiveMicro session is fairly similar to the one spent in deploying a VM on OpenStack and so non-negligible. On the other hand, if we cache a telepathology session, its bootstrap, *i.e.*, resume time, is more than 200 times faster. These results validate our approach of maintaining a pool of resources already deployed and turned them on when necessary.

TABLE II: Startup latency in case of a pool of VMs already deployed, and case when VM is created for each client.

	Login	Get Image
Pool	0.447 s	0.450 s
On demand	3.940 s	120.062 s

Moreover, we analyze the load on the physical machine, for the bootstrap of a session, to see how this can impact on system performance. In particular, we evaluate memory and CPU usage. Figure 8a shows how the creation of a new LiveMicro session does not extensively impact the used memory (RAM), in a classic case of two-users session deployment; this means that two pathologists in different locations are both connected to the same remote microscope. We note that for each VM hosting a telepathology session, the RAM usage increases only of 2 MB, and the memory is used until the VM is destroyed, when the resources are hence released. Considering the total RAM on the physical machine was about 250 GB, we can state that creating and maintaining a telepathology session with LiveMicro is negligible, as we measured an overall consumption of less than 1%. Figure 8b shows, instead, the CPU usage in the same conditions as the prior experiment: a two-users telepathology session. In this case, we note how, except for an initial peak, the deployed VM does not particularly impact the CPU load: the usage never exceeds 6%.

Given our resource usage findings, we conclude that LiveMicro can run smoothly if deployed on a fairly powerful server capable of running the two main OpenStack services, i.e., the compute service and the networking service (see Section IV), plus an additional one to handle each telepathology session. In fact, given a well-dimensioned pool, the startup time for obtaining a VM is reasonable for the user, that is asked to wait for less than 1 second for the login request.



(a) (b) (c) (d)

Fig. 8: (a) OpenStack memory usage during the startup of 2 VMs. For each VM creation we notice an increment. (b) OpenStack CPU usage, during the startup of 2 VMs. We notice that the CPU is just partly affected. (c) The samples are displayed in ascending order of size. For each image, we measured the image retrieval latency, in case it is on a local file or on a database. The fact that images are not stored locally does not affect performance always in the same way. (d) Waiting time for an increasing number of users accessing the system concurrently.

TABLE III: Time required for LiveMicro requests to access microscope.

	10 m	1 Km	960 Km
Network Latency	0.92 ms	1.64 ms	0.14 s
Configuration	0.34 s	0.78 s	1.32 s
Stage Move	0.22 s	0.65 s	1.52 s
Z-stack Move	0.49 s	0.99 s	1.98 s
Light control	0.19 s	0.57 s	1.10 s

To assess the system performance from the user perspective, in Figure 8c we report the time to retrieve samples of increasing size when already in the memory of the VM (best case) or in the database (worst case). The results confirm the validity of our approach to store locally the image so that consecutive requests benefit from the cache layer provided by VMs. Besides, we can note how the time to access the database is affected by the image size and format, as well as by other factors that can vary from case to case, e.g., caching policies of the database management system. It is relevant to note that the latency shown in the figure is the time to collect and display the whole image, a rare scenario that usually happens just once at the beginning. In practice, the user operates on a small portion of the sample to analyze in more detail shapes and colors, leading to a reduced latency perceived.

C. Real-microscope evaluation

In this section, we show our latency evaluation results on an Olympus IX81 [29] microscope connected to a network interface through our Plugin program. Table III reports the average experienced latency over 20 runs considering four different types of requests for a microscope located at Saint Louis University, varying the distance between the client and the microscope from a few meters to 960 Km. As a baseline, we also report the network latency, which is measured through the *ping* tool, and it indicates the time to send a (small) packet over the network.

Our results confirm that the time to perform an action even on a real microscope is lower than the standard threshold for enabling a real-time system [50]. Only for the most remote location (960 Km), the latency increases to values that might be too high, but still lower than other similar solutions [51].

We can observe that for very distant locations, other arrangements should be considered for optimal user experience. For these distances, the problem is indeed the possible congestion, that can be tackled both via bandwidth reservation among separate sites, such as Virtual Private Networks, and selecting different congestion control protocols that are more suited for latency-critical applications. Moreover, other solutions can be designed alongside the Internet Service Provider (ISP) for a specific Quality of Service (QoS), which usually entails a payment. In the past, the interaction with the ISP was necessary to provide the service, e.g., in [52], where the Norwegian Telecom reserves enough bandwidth for the communication. Although nowadays the reservation is not a pre-requisite, for optimal service interactivity, this option may be valuable. Finally, when we process digital slides offline, we show that even scarce but well-managed (CPU and memory) resources can be adequate for an effective telepathology.

VII. Conclusion

Telepathology, the practice of pathology at long distance by pathologists has been around since 1986 but never took off due to poor performance and the lack of usability. This paper presents a novel system aiming to advance significantly the field of telepathology by augmenting live remote microscope sessions with the computational power of (cutting) edge computing technologies. Such a platform allows a team of pathologists to access, control, and process, simultaneously, a remotely located (real or emulated) microscope using merely a (mobile) web browser. We describe the architecture of our system and disclose its potentials to improve the field of medical diagnosis in critical situations, for example under an active surgery, where a quick diagnosis is vital but experts are locally unavailable.

Furthermore, our results show how an edge computing-empowered microscope provides additional benefits such as performing application-specific image processing and speeding up image transmission time. Such processing involves, for example, tumoral cell identification to accelerate pathology diagnosis (currently expressed cells are manually counted) and to support continuous education in pathology.

Future work could focus on both aspects of this cyber-human system. From the system side, researchers could focus on enabling telepathology also on longer and congested paths using novel latency-sensitive protocols, or applying more complex pattern recognition algorithms on digital slides. From the human point of view, future work could involve assessments of Quality of Experience of LiveMicro in biomedical education.

ACKNOWLEDGMENT

This work has been partially supported by NSF Awards CNS1647084, CNS1836906, and CNS1908574.

REFERENCES

[1] S. Al-Janabi, A. Huisman, and P. J. Van Diest, "Digital pathology: current status and future perspectives," *Histopathology*, vol. 61, no. 1, pp. 1–9, 2012.
[2] P. W. Hamilton, Y. Wang, and S. J. McCullough, "Virtual microscopy and digital pathology in training and education," *Apmis*, vol. 120, no. 4, pp. 305–315, 2012.
[3] B. Têtu *et al.*, "Whole-slide imaging-based telepathology in geographically dispersed healthcare networks. the eastern québec telepathology project," *Diagnostic Histopathology*, vol. 20, no. 12, pp. 462–469, 2014.
[4] A. Sacco, F. Esposito, P. Okorie, and G. Marchetto, "Livemicro: An edge computing system for collaborative telepathology," in *2nd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.
[5] Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, and H. Greenspan, "Chest pathology detection using deep learning with non-medical training," in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2015, pp. 294–297.
[6] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, "Brain tumor segmentation using convolutional neural networks in mri images," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1240–1251, 2016.
[7] R. Rouhi, M. Jafari, S. Kasaei, and P. Keshavarzian, "Benign and malignant breast tumors classification based on region growing and cnn segmentation," *Expert Systems with Applications*, vol. 42, no. 3, pp. 990–1002, 2015.
[8] R. S. Weinstein *et al.*, "Telepathology overview: from concept to implementation," *Human pathology*, vol. 32, no. 12, pp. 1283–1299, 2001.
[9] R. S. Weinstein, A. R. Graham *et al.*, "Overview of telepathology, virtual microscopy, and whole slide imaging: prospects for the future," *Human pathology*, vol. 40, no. 8, pp. 1057–1069, 2009.
[10] K. Kayser, M. Oberholzer, G. WeißZe, I. WeißZe, and H. v. Eberstein, "Long distance image transfer: First results of its use in histopathological diagnosis," *APMIS*, vol. 99, no. 7-12, pp. 808–814, 1991.
[11] J. Linder, "Overview of digital imaging in pathology. the fifth wave," *American journal of clinical pathology*, vol. 94, no. 4 Suppl 1, pp. S30–4, 1990.
[12] R. Bashshur, *Telemedicine; explorations in the use of telecommunications in health care*. Charles C. Thomas Publisher, 1975.
[13] P. Schwarzmann, J. Schmid, C. Schnörr, G. Straessle, and S. Witte, "Telemicroscopy stations for telepathology based on broadband and ISDN connections," *Archives d'anatomie et de cytologie pathologiques*, vol. 43, no. 4, pp. 209–215, 1995.
[14] T. S. Winokur *et al.*, "A prospective trial of telepathology for intraoperative consultation (frozen sections)," *Human pathology*, vol. 31, no. 7, pp. 781–785, 2000.
[15] P. Bartels, "Automated primary screening devices. expectations for the next generation," *Acta cytologica*, vol. 44, no. 5, pp. 703–708, 2000. [Online]. Available: https://doi.org/10.1159/000328552
[16] A. Afework *et al.*, "Digital dynamic telepathology–the virtual microscope." in *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 1998, p. 912.
[17] M. R. Descour *et al.*, "Toward the development of miniaturized imaging systems for detection of pre-cancer," *IEEE Journal of Quantum Electronics*, vol. 38, no. 2, pp. 122–130, 2002.
[18] R. K. Kumar, G. M. Velan, S. O. Korell, M. Kandara, F. R. Dee, and D. Wakefield, "Virtual microscopy for learning and assessment in pathology," *The Journal of Pathology: A Journal of the Pathological Society of Great Britain and Ireland*, vol. 204, no. 5, pp. 613–618, 2004.
[19] D. U. Ekong and P. Fontelo, "Prototype telepathology solutions that use the raspberry pi and mobile devices," in *Global Humanitarian Technology Conference (GHTC), 2017 IEEE*. IEEE, 2017, pp. 1–4.
[20] R. Dudas, C. VandenBussche, A. Baras, S. Z. Ali, and M. T. Olson, "Inexpensive telecytology solutions that use the raspberry pi and the iphone," *Journal of the American Society of Cytopathology*, vol. 3, no. 1, pp. 49–55, 2014.
[21] R. K. Richard Weinberg, "Digital tele-microscopy in support of teaching biology." [Online]. Available: http://grantome.com/grant/NSF/CNS-1451220
[22] M. Berman, C. Elliott, and L. Landweber, "Geni: Large-scale distributed infrastructure for networking and distributed systems research," in *2014 IEEE Fifth International Conference on Communications and Electronics (ICCE)*. IEEE, 2014, pp. 156–161.
[23] Acquiring from GigE Vision Cameras with Vision Acquisition Software. [Online]. Available: http://www.ni.com/white-paper/5651/en/
[24] R. Lebre, R. Jesus, P. Nunes, and C. Costa, "Collaborative framework for a whole-slide image viewer," in *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE, 2019, pp. 221–224.
[25] C. Alvarez, G. Corredor, D. Giraldo, and E. Romero, "Tele-pathology: A use case in colombia," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE, 2019, pp. 1417–1421.
[26] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.
[27] L. B. Wilson, "A method for the rapid preparation of fresh tissues for the microscope." *Journal of the American Medical Association*, vol. 45, no. 23, pp. 1737–1737, 1905.
[28] OpenSeadragon. [Online]. Available: http://openseadragon.github.io/
[29] Olympus IX81 microscope, [Online]. Available: https://www.biocompare.com/19419-Inverted-Microscopes/399623-IX81-Inverted-Microscope/.
[30] FFmpeg documentation, [Online]. Available: https://www.ffmpeg.org/.
[31] WebRTC free and open project. [Online]. Available: https://webrtc.org/
[32] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "Openstack: toward an open-source solution for cloud computing," *International Journal of Computer Applications*, vol. 55, no. 3, pp. 38–42, 2012.
[33] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.
[34] S. Tilkov and S. Vinoski, "Node. js: Using javascript to build high-performance network programs," *IEEE Internet Computing*, vol. 14, no. 6, pp. 80–83, 2010.
[35] gRPC framework. [Online]. Available: https://grpc.io/docs/
[36] PyramidIO. [Online]. Available: https://github.com/usnistgov/pyramidio/
[37] S. Paluru and J. I. Epstein, "Does the distance between tumor and margin in radical prostatectomy specimens correlate with prognosis: relation to tumor location," *Human pathology*, vol. 56, pp. 11–15, 2016.
[38] M. Macenko *et al.*, "A method for normalizing histology slides for quantitative analysis," in *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. IEEE, 2009, pp. 1107–1110.
[39] B. E. Bejnordi *et al.*, "Stain specific standardization of whole-slide histopathological images," *IEEE transactions on medical imaging*, vol. 35, no. 2, pp. 404–415, 2015.
[40] Tumor Challenge. [Online]. Available: http://tupac.tue-image.nl/
[41] F. Ciompi *et al.*, "The importance of stain normalization in colorectal tissue classification with convolutional networks," in *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*. IEEE, 2017, pp. 160–163.
[42] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
[43] S. Suzuki *et al.*, "Topological structural analysis of digitized binary images by border following," *Computer vision, graphics, and image processing*, vol. 30, no. 1, pp. 32–46, 1985.
[44] G. Y. Tang, "A discrete version of green's theorem," *IEEE transactions on pattern analysis and machine intelligence*, no. 3, pp. 242–249, 1982.
[45] L. F. Williams Jr, "A modification to the half-interval search (binary search) method," in *Proceedings of the 14th annual Southeast regional conference*. ACM, 1976, pp. 95–101.
[46] R. Ricci, E. Eide, and C. Team, "Introducing cloudlab: Scientific infrastructure for advancing cloud architectures and applications," *; login:: the magazine of USENIX & SAGE*, vol. 39, no. 6, pp. 36–38, 2014.
[47] Digital Slide Archive (DSA). [Online]. Available: https://cancer.digitalslidearchive.org/
[48] 2018 Data Science Bowl. [Online]. Available: https://www.kaggle.com/c/data-science-bowl-2018
[49] F. G. Zanjani, S. Zinger *et al.*, "Stain normalization of histopathology images using generative adversarial networks," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE, 2018, pp. 573–577.
[50] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5g: Ran, core network and caching solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, 2018.
[51] P. Schulz *et al.*, "Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, 2017.
[52] I. Nordrum *et al.*, "Remote frozen section service: a telepathology project in northern norway," *Human pathology*, vol. 22, no. 6, pp. 514–518, 1991.