CAMP - RT

A Patient Similarity-Based Method for Predicting Radiation Dosage
in Head and Neck Radiation Oncology

Peter Hanula
UIN 660468116
Fall 2018

Advisor: **G. Elisabeta Marai, PhD**
Reader: Andrew E. Johnson, PhD

The University of Illinois at Chicago
Electronic Visualization Laboratory

Table of Contents:

ABSTRACT

I present an automated method for computing the spatial similarity of tumor location with respect to organs at risk between multiple head and neck cancer patients and ranking the patients accordingly. I use this method and resulting metric to predict, for each patient, a radiation dose distribution across organs in the head and neck. Furthermore, I go over the design and implementation of a visual analysis tool used to help validate the method.

The method uses as input DICOM (Digital Imaging and Communications in Medicine) files [1] provided by our collaborators at MD Anderson Cancer Center. We preprocess these files to extract primary data components and translate them into an efficient format for the automated method and web-based application to read in. The method is written in Python, and utilizes the Structural SIMilarity (SSIM) [2] algorithm for computing similarities. A JSON (JavaScript Object Notation) file [4] is generated containing all patients, and for every patient a dose prediction and ordered list of most similar patients is produced. The web-based GUI (Graphical User Interface) reads in the JSON file and lets users visually explore the data, analyze the method, and gather insights for possible radiation therapy plans.

The resulting method and interface, CAMP-RT (Correlations Across Multiple Patients in Radiation Therapy) is continuously being evaluated by domain experts from the MD Anderson Cancer Center. I report their qualitative feedback regarding the method and interface. I further evaluate quantitatively the results of the prediction using a dataset of 101 head-and-neck cancer patients. The results indicate the method has good predictive capabilities.

1. INTRODUCTION

Radiation therapy is one of the most common treatments for cancer, using high-energy radiation that can kill cancer cells and shrink tumors. However, the radiation can also damage normal cells and organs, leading to various side effects. Some examples include dry mouth, mouth and gum sores, difficulty swallowing, stiffness in the jaw, tooth decay, increased risk of stroke, and peripheral neuropathy. Side effects develop when healthy organs are located in the radiation field during radiation therapy. These organs are known as OARs (Organs At Risk). The goal of the automated method that I am presenting is to allow radiologists to optimize the radiation therapy plan in a shorter time frame, while minimizing pain and side effects to healthy organs at risk.

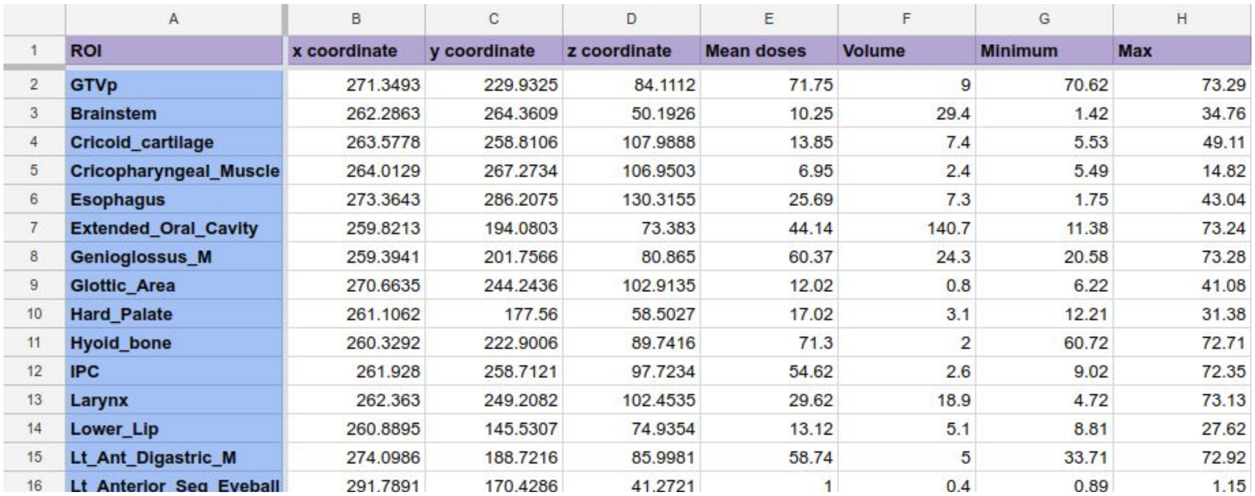
2. METHODS

The method uses as input DICOM files provided by our collaborators at MD Anderson Cancer Center. We preprocess these files to extract primary data components and translate them into an efficient format for the automated method and web-based application to read in. The method is written in Python, and utilizes the SSIM algorithm for computing similarities. A JSON file is generated containing all patients, and for every patient a dose prediction and ordered list of most similar patients is produced. The web-based GUI reads in the JSON file and lets users visually explore the data, analyze the method, and gather insights for possible radiation therapy plans. I describe below in more detail each component.

2.1 Data

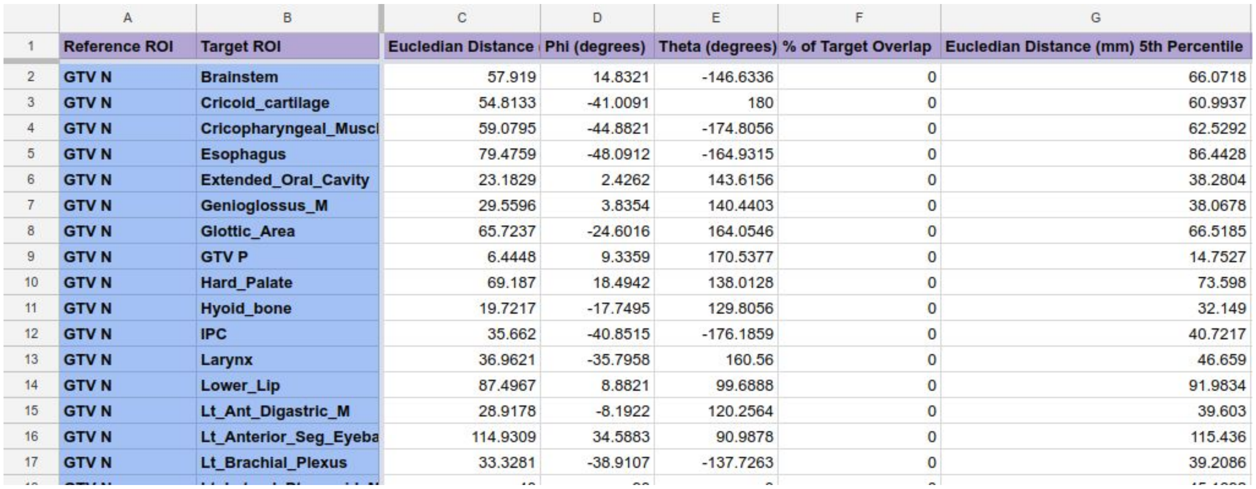
Most of the patient data originates from DICOM files. Our collaborators at MD Anderson Cancer Center extract only the necessary data components, which are then organized and shared with us through a large group of CSV (Comma-Separated Values) files. This process naturally fits into their workflow, and dramatically reduces file sizes. The current dataset contains 101 patients. Two CSV files are produced for every patient. The first file *id_centroid_meandoses.csv* lists the patient's organs and tumors with attributes **mean**, **minimum**, and **max doses** (GY), along with the **x**, **y**, **z coordinates** and **volume** (cc). Each patient has data for around 50 organs in the head and neck regions, including the primary GTVp and secondary GTVn tumors (Gross Tumor Volume). Doctors want to eliminate tumors while trying to avoid damage to healthy organs at risk. Organs located closer to the tumors receive higher levels of radiation.

The second file *id_distances.csv* lists the **organ to organ/tumor distances** (mm) for every possible pair combination.



	A	B	C	D	E	F	G	H
1	ROI	x coordinate	y coordinate	z coordinate	Mean doses	Volume	Minimum	Max
2	GTVp	271.3493	229.9325	84.1112	71.75	9	70.62	73.29
3	Brainstem	262.2863	264.3609	50.1926	10.25	29.4	1.42	34.76
4	Cricoid_cartilage	263.5778	258.8106	107.9888	13.85	7.4	5.53	49.11
5	Cricopharyngeal_Muscle	264.0129	267.2734	106.9503	6.95	2.4	5.49	14.82
6	Esophagus	273.3643	286.2075	130.3155	25.69	7.3	1.75	43.04
7	Extended_Oral_Cavity	259.8213	194.0803	73.383	44.14	140.7	11.38	73.24
8	Genioglossus_M	259.3941	201.7566	80.865	60.37	24.3	20.58	73.28
9	Glottic_Area	270.6635	244.2436	102.9135	12.02	0.8	6.22	41.08
10	Hard_Palate	261.1062	177.56	58.5027	17.02	3.1	12.21	31.38
11	Hyoid_bone	260.3292	222.9006	89.7416	71.3	2	60.72	72.71
12	IPC	261.928	258.7121	97.7234	54.62	2.6	9.02	72.35
13	Larynx	262.363	249.2082	102.4535	29.62	18.9	4.72	73.13
14	Lower_Lip	260.8895	145.5307	74.9354	13.12	5.1	8.81	27.62
15	Lt_Ant_Digastric_M	274.0986	188.7216	85.9981	58.74	5	33.71	72.92
16	Lt_Anterior_Seg_Eyeball	291.7891	170.4286	41.2721	1	0.4	0.89	1.15

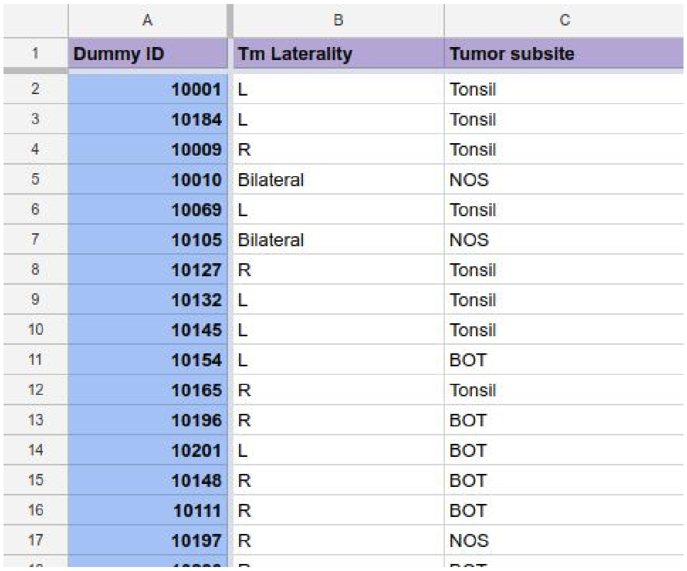
Example of id_centroid_meandoses.csv



	A	B	C	D	E	F	G
1	Reference ROI	Target ROI	Eucledian Distance	Phi (degrees)	Theta (degrees)	% of Target Overlap	Eucledian Distance (mm) 5th Percentile
2	GTV N	Brainstem	57.919	14.8321	-146.6336	0	66.0718
3	GTV N	Cricoid_cartilage	54.8133	-41.0091	180	0	60.9937
4	GTV N	Cricopharyngeal_Muscl	59.0795	-44.8821	-174.8056	0	62.5292
5	GTV N	Esophagus	79.4759	-48.0912	-164.9315	0	86.4428
6	GTV N	Extended_Oral_Cavity	23.1829	2.4262	143.6156	0	38.2804
7	GTV N	Genioglossus_M	29.5596	3.8354	140.4403	0	38.0678
8	GTV N	Glottic_Area	65.7237	-24.6016	164.0546	0	66.5185
9	GTV N	GTV P	6.4448	9.3359	170.5377	0	14.7527
10	GTV N	Hard_Palate	69.187	18.4942	138.0128	0	73.598
11	GTV N	Hyoid_bone	19.7217	-17.7495	129.8056	0	32.149
12	GTV N	IPC	35.662	-40.8515	-176.1859	0	40.7217
13	GTV N	Larynx	36.9621	-35.7958	160.56	0	46.659
14	GTV N	Lower_Lip	87.4967	8.8821	99.6888	0	91.9834
15	GTV N	Lt_Ant_Digastric_M	28.9178	-8.1922	120.2564	0	39.603
16	GTV N	Lt_Anterior_Seg_Eyeba	114.9309	34.5883	90.9878	0	115.436
17	GTV N	Lt_Brachial_Plexus	33.3281	-38.9107	-137.7263	0	39.2086

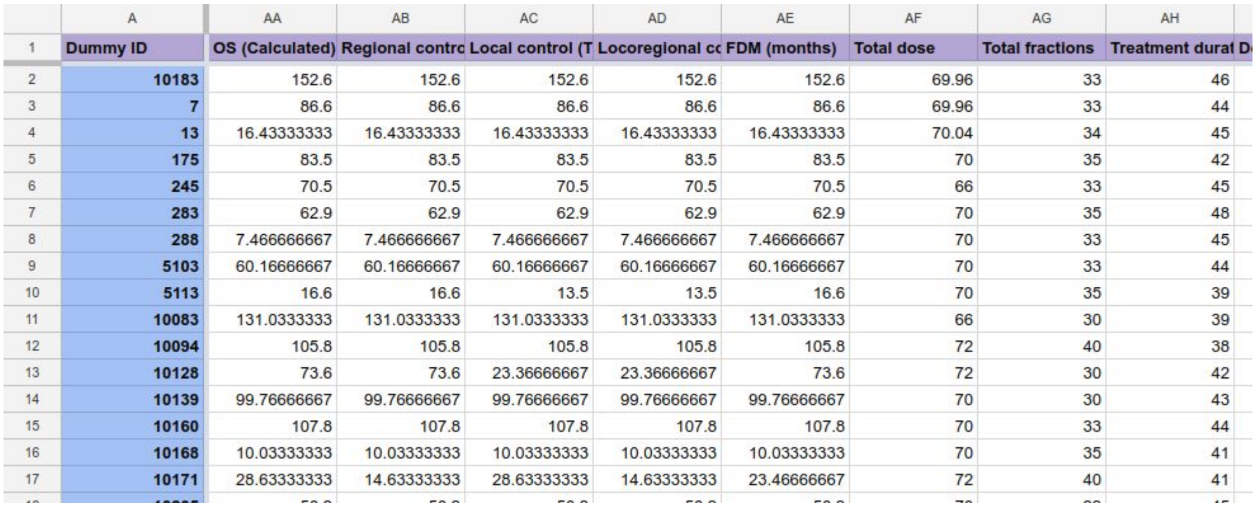
Example of id_distances.csv

Additional data is domain-specific implicit knowledge and meta-data located in our domain experts' heads. We extract and encode some of this data through multiple interactions with our collaborators and repeated feedback sessions. For example, there is a *laterality.csv* file listing the **tumor laterality** (left, right, bilateral) and **tumor subsite** (BOT, Tonsil, Soft Palate, Pharyngeal Wall, GPS, NOS) of each patient. There is an additional clinical file listing 40 additional attributes for each patient, although our method only incorporates the **total dose** (GY) attribute. The total dose corresponds to the total radiation a patient has received during treatment, or will receive if treatment hasn't started.



	A	B	C
1	Dummy ID	Tm Laterality	Tumor subsite
2	10001	L	Tonsil
3	10184	L	Tonsil
4	10009	R	Tonsil
5	10010	Bilateral	NOS
6	10069	L	Tonsil
7	10105	Bilateral	NOS
8	10127	R	Tonsil
9	10132	L	Tonsil
10	10145	L	Tonsil
11	10154	L	BOT
12	10165	R	Tonsil
13	10196	R	BOT
14	10201	L	BOT
15	10148	R	BOT
16	10111	R	BOT
17	10197	R	NOS

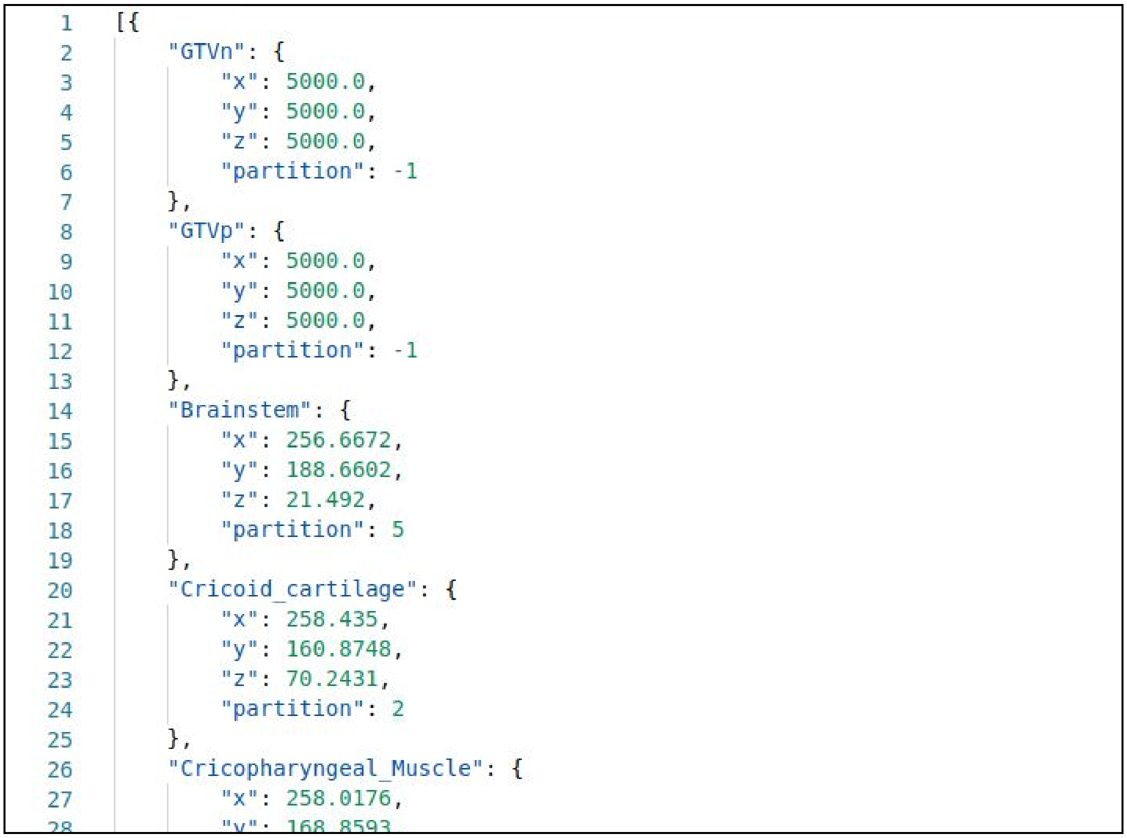
Example of laterality.csv



	A	AA	AB	AC	AD	AE	AF	AG	AH
1	Dummy ID	OS (Calculated)	Regional contro	Local control (T	Locoregional cc	FDM (months)	Total dose	Total fractions	Treatment durat
2	10183	152.6	152.6	152.6	152.6	152.6	69.96	33	46
3	7	86.6	86.6	86.6	86.6	86.6	69.96	33	44
4	13	16.43333333	16.43333333	16.43333333	16.43333333	16.43333333	70.04	34	45
5	175	83.5	83.5	83.5	83.5	83.5	70	35	42
6	245	70.5	70.5	70.5	70.5	70.5	66	33	45
7	283	62.9	62.9	62.9	62.9	62.9	70	35	48
8	288	7.466666667	7.466666667	7.466666667	7.466666667	7.466666667	70	33	45
9	5103	60.16666667	60.16666667	60.16666667	60.16666667	60.16666667	70	33	44
10	5113	16.6	16.6	13.5	13.5	16.6	70	35	39
11	10083	131.0333333	131.0333333	131.0333333	131.0333333	131.0333333	66	30	39
12	10094	105.8	105.8	105.8	105.8	105.8	72	40	38
13	10128	73.6	73.6	23.36666667	23.36666667	73.6	72	30	42
14	10139	99.76666667	99.76666667	99.76666667	99.76666667	99.76666667	70	30	43
15	10160	107.8	107.8	107.8	107.8	107.8	70	33	44
16	10168	10.03333333	10.03333333	10.03333333	10.03333333	10.03333333	70	35	41
17	10171	28.63333333	14.63333333	28.63333333	14.63333333	23.46666667	72	40	41

Example of clinical CSV file

Furthermore, following discussions with our collaborators at MD Anderson Cancer Center, a *organAtlas.json* file was created with a list of organs they are most interested in. The atlas currently contains **47 organs**, including the primary GTVp and secondary GTVn tumors. The organs are partitioned into 6 groups: **Oral Cavity & Jaw**, **Throat**, **Salivary Glands**, **Eyes**, **Brainstem & Spinal Cord**, and **Other**.



```
[{
    "GTVn": {
        "x": 5000.0,
        "y": 5000.0,
        "z": 5000.0,
        "partition": -1
    },
    "GTVp": {
        "x": 5000.0,
        "y": 5000.0,
        "z": 5000.0,
        "partition": -1
    },
    "Brainstem": {
        "x": 256.6672,
        "y": 188.6602,
        "z": 21.492,
        "partition": 5
    },
    "Cricoid_cartilage": {
        "x": 258.435,
        "y": 160.8748,
        "z": 70.2431,
        "partition": 2
    },
    "Cricopharyngeal_Muscle": {
        "x": 258.0176,
        "y": 168.8593
```

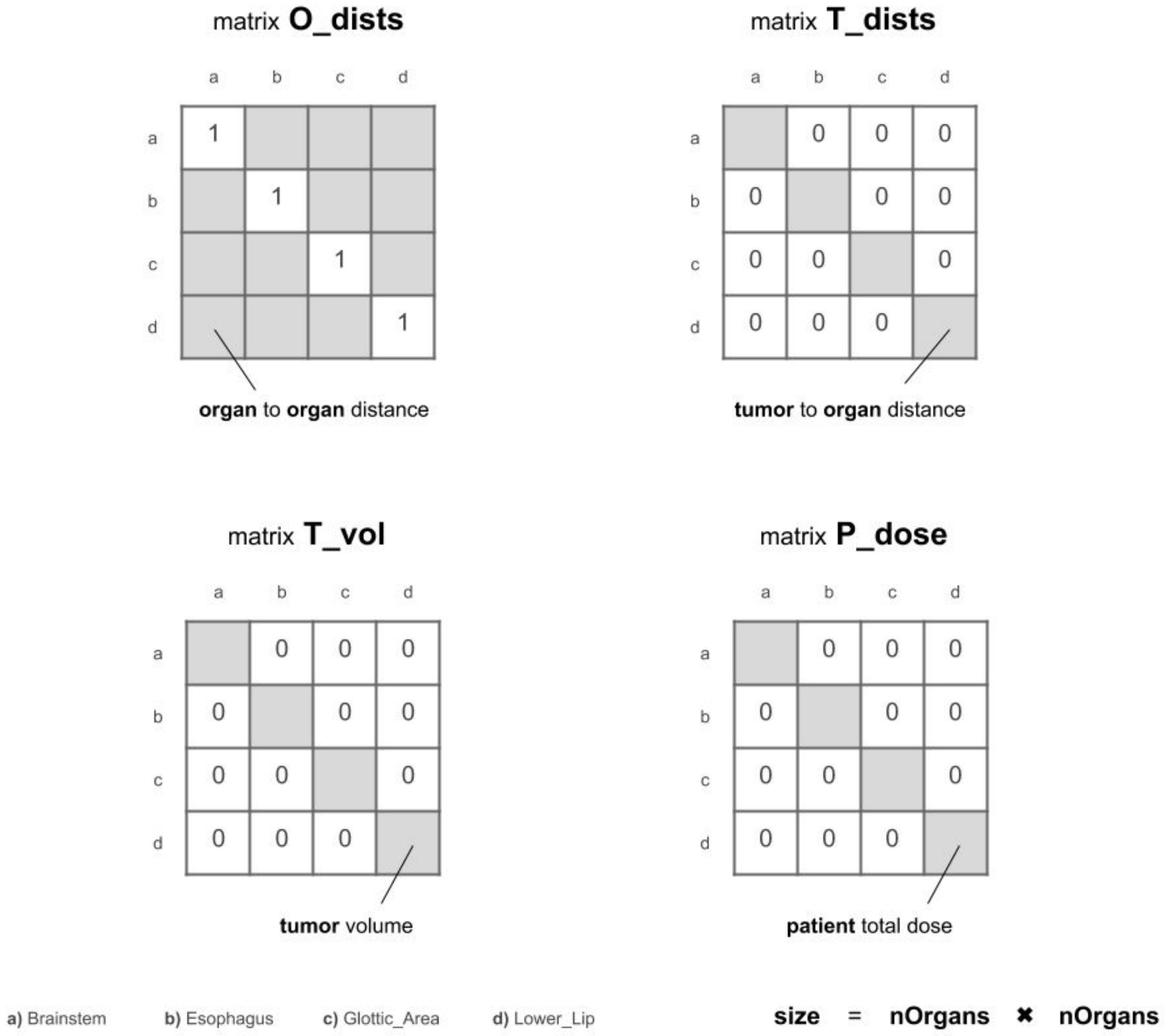
Example of organAtlas.json

Data Integration and Representation

The Python script starts by searching through the patients folder, returning a list of all the CSV pathnames associated with the patients. The script then iterates through the list of files, reading in and parsing one CSV file at a time. Since there are two CSV files associated with each patient, a patient entry is either being created or updated. For each new patient encountered, a OrderedDict Collection instance is returned to store the patient's information, and then the OrderedDict is appended to a list of patients. The **ID** and **name** can be retrieved from either file. The program also assigns **"internal" IDs** to ensure each patient has a unique identifier. From the *id_centroid_meandoses.csv* file, organ and tumor information is parsed and organized as a OrderedDict of OrderedDicts, stored in the variable **organData**. Each organ and tumor contains **x**, **y**, **z**, **volume**, **meanDose**, **minDose**, and **maxDose** attributes.

From the *id_distances.csv* file, two matrices are then created and populated, along with two boolean variables **hasGTVp** and **hasGTVn**. These boolean variables are used internally when populating data relating to tumors. The rows and columns of a matrix map to individual organs. The size of the matrices depend on how many organs are included in the organ atlas, in this case 47 x 47.

In the first matrix **O_dists**, each cell contains the organ to organ distance for that row, column pair combination. The diagonal of the matrix is populated with 1's. In the second matrix **T_dists**, each cell of the diagonal contains the tumor to organ distance for that respective organ. The rest of the cells are populated with 0's.



Matrix structures for storing patient data. In practice, the matrices are scaled to a size of 47 x 47.

After all the individual patient files have been parsed, a third matrix is generated for each patient, **T_vol**. The diagonal of the matrix is populated with the primary GTVp tumor volume, and the remaining cells are populated with 0's.

Then, a fourth matrix is generated for each patient, named **P_dose**, which relies on the total dose attribute found in the clinical csv file. The file is read and parsed, the patient ID's are cross referenced with the patient data already stored, and then the patient data is updated with a new attribute **total_Dose**. The diagonal of the fourth matrix is populated with the patient's total dose, and the remaining cells are populated with 0's.

Finally, the *laterality.csv* file is read and parsed, the patient ID's are cross referenced with the patient data already stored, and then the patient data is updated with two new attributes **laterality** and **tumor_Subsite**.

Data Processing

M_dists = O_dists □ T_dists

M_vol = O_dists □ T_vol

M_dose = O_dists □ P_dose

The data needs to be processed before it can be ready for the similarity algorithm. The four matrices produced are used to compute three new matrices. The distance matrix **M_dists** is calculated by taking the dot product of **O_dists** and **T_dists**. The volume matrix **M_vol** is calculated by taking the dot product of **O_dists** and **T_vol**. The dose matrix **M_dose** is calculated by taking the dot product of **O_dists** and **P_dose**. With the way the four matrices are structured and populated, taking the dot product distributes the right-hand matrix (T_dists, T_vol, or P_dose) into the left-hand matrix (O_dists), resulting in similarity scores that cover a greater range. The whole data pipeline for the 101 patients, from reading in the distributed data sources to computing the dot products, executes on average in five seconds.

2.2 Algorithm: Structural SIMilarity (SSIM)

ssim_dists = SSIM (currP[‘M_dists’] , nextP[‘M_dists’])

ssim_vol = SSIM (currP[‘M_vol’] , nextP[‘M_vol’])

ssim_dose = SSIM (currP[‘M_dose’] , nextP[‘M_dose’])

score = (ssim_dists + ssim_vol + ssim_dose + laterality) / 4

The Structural SIMilarity index is a method for measuring the similarity between two images, in our case matices. We are using a Matlab implementation of the algorithm [3], exported as a Python package to work seamlessly with the current workflow. The Matlab code was modified to accept matrices as input instead of image files.

In the Python script, the call to the SSIM algorithm sits inside a double for loop, where the outer loop sets the current patient and the inner loop iterates through all the patients. The SSIM algorithm is called three times when comparing the target patient to a particular patient, computing **ssim_dists**, **ssim_vol**, and **ssim_dose**. A final score is determined by adding the **ssim_dists**, **ssim_vol**, **ssim_dose**, and **laterality** values, and then normalizing the sum to a common scale ranging from 0 to 1.

The inner loop iterates through all the patients and calculates the similarity scores for that set of patients, appending the IDs and scores as tuples to a list. The list is sorted by the similarity score. One iteration of the outer loop takes, on average, 10 seconds to execute. The ranking order and similarity scores are extracted and stored in variables **similarity_ssim** and **scores_ssim**, respectively. The patient’s data is updated to include these variables. The process is then repeated for the remaining patients, set through the outer for loop.

similarity_ssim = [24, 44, 21, 35, 42, 25, 52, 11, …]

scores_ssim = [1.0, 0.989, 0.981, 0.961, 0.960, 0.951, 0.944, 0.922, …]

Example of data extracted for a certain patient.

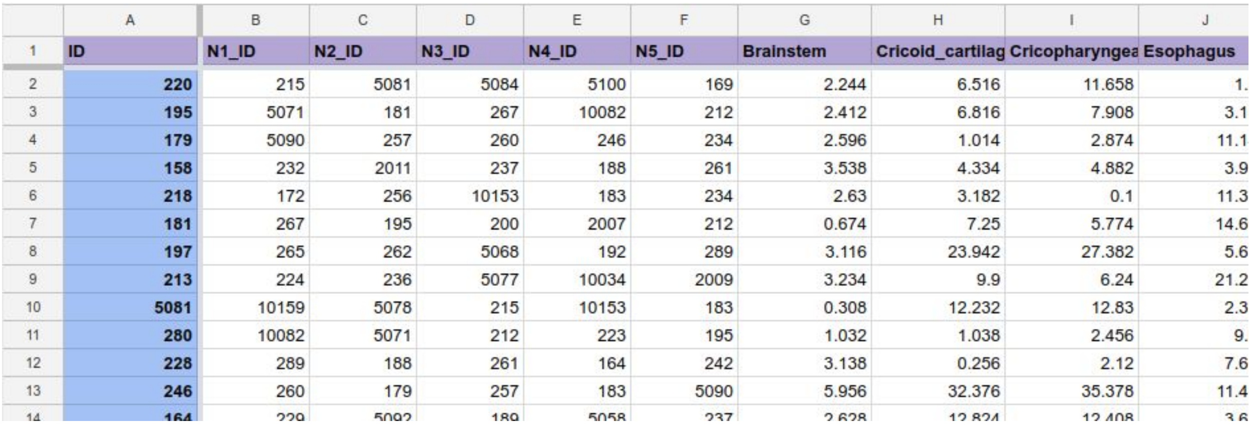
Notice that radiation dose information per organs is not utilized in the similarity algorithm (just the total dose a patient received or will receive). This was a deliberate choice. In a real use case, the target patient would be a new patient who hasn't started radiation therapy. The amount of radiation a certain organ will potentially receive is not known at that point and it is difficult to determine. The automated method was designed to use known or easily derived data points to determine how similar two patients are, and then estimate new data points.

Our hypothesis is that the more similar two patients are, the more likely it is that they will receive similar radiation therapy treatments. To test this hypothesis, we use the similarity method to identify the five most similar patients for any given patient in our dataset. We then use the known radiation dosage distribution across the five most similar patients to predict the radiation dosage distribution (the amount of radiation per each organ) of the given patient. We validate the prediction against the actual radiation dosage distribution for that patient.

Dose Prediction

Once all the similarity scores have been computed, dose predictions can be made. For each patient, the top five most similar patients are used to predict a dose distribution across all organs. Since these patients have already received radiation therapy, it is known what level of radiation each organ has received. Currently the top five most similar patients are averaged on a per organ basis to determine the predicted dose for that organ. The patient's data is updated to include the dose predictions.

Before the automated method finishes executing, it writes to disk the file *differences.csv*, where each row corresponds to a patient. The columns consist of the patient's ID, followed by the IDs of the top five most similar patients, and then every organ difference. A organ difference for a particular organ is determined by subtracting the predicted dose value by the actual dose value received, and then taking the absolute value. This is possible because our patient dataset consists of patients that all received radiation therapy treatment, so the actual dose values for each organ is known. In the case where data is missing, the difference value defaults to "-1" and the findings are reported to our collaborators. By comparing the predicted dose with the actual dose, the automated method can be analyzed and validated. The *differences.csv* is used for further statistical analysis and k-fold cross-validation. The goal of cross-validating the data is to test the method's ability to predict new data that was not used in estimating it.



	A	B	C	D	E	F	G	H	I	J
1	**ID**	**N1_ID**	**N2_ID**	**N3_ID**	**N4_ID**	**N5_ID**	**Brainstem**	**Cricoid_cartilag**	**Cricopharyngea**	**Esophagus**
2	**220**	215	5081	5084	5100	169	2.244	6.516	11.658	1.
3	**195**	5071	181	267	10082	212	2.412	6.816	7.908	3.1
4	**179**	5090	257	260	246	234	2.596	1.014	2.874	11.1
5	**158**	232	2011	237	188	261	3.538	4.334	4.882	3.9
6	**218**	172	256	10153	183	234	2.63	3.182	0.1	11.3
7	**181**	267	195	200	2007	212	0.674	7.25	5.774	14.6
8	**197**	265	262	5068	192	289	3.116	23.942	27.382	5.6
9	**213**	224	236	5077	10034	2009	3.234	9.9	6.24	21.2
10	**5081**	10159	5078	215	10153	183	0.308	12.232	12.83	2.3
11	**280**	10082	5071	212	223	195	1.032	1.038	2.456	9.
12	**228**	289	188	261	164	242	3.138	0.256	2.12	7.6
13	**246**	260	179	257	183	5090	5.956	32.376	35.378	11.4
14	**164**	229	5092	189	5058	237	2.628	12.824	12.408	3.6

Differences.csv [12]. The spreadsheet link can be found in the references section.

Further research in predicting the dose distribution is currently being conducted, and the process will be refined in the following semesters.

Implementation

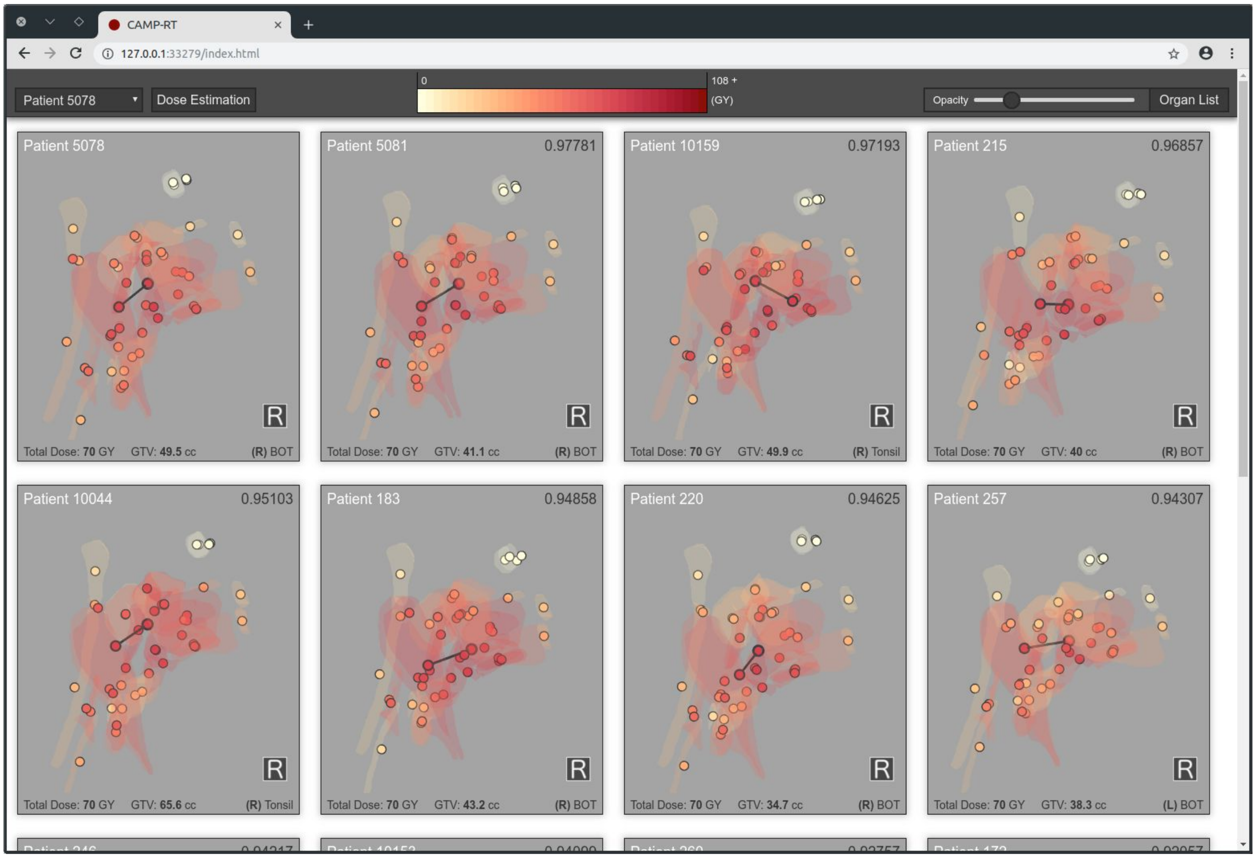
The automated method is written in Python v2.7.15 [5]. The numpy v1.15.4 module [6] is used to represent each patient as a series of matrices for further data processing. The pySSIM v1.0 package was originally written in Matlab, and is used to compute a similarity score between two patients. I exported the code as a Python module to seamlessly integrate with our workflow. However, this requires the MATLAB Runtime [7] to be installed on the machine running the Python script. The matlab v0.1 [8] module is used to bridge the pySSIM module and convert data to the appropriate format. Module glob v2.5 [9] is used to find and return a list of all relevant patient pathnames, while csv v2.3 [10] and json v2.6 [11] are used to import and export files in the appropriate format. The Python script is executed using the command “python2.7 CAMPRTdata.py patients/”. “patients/” is a command-line argument specifying the folder location where patient data is stored.

2.3 Visual Analysis Tool

The Python script described above generates a second file *patients.json*, which contains all the patient data and is read in by the visual analysis tool. Before this JSON file is created the patient data, a list of OrderedDicts, is cleaned. All matrices that have been created are deleted, because these were only necessary when applying the similarity algorithm. With Python, a list of OrderedDicts can simply be dumped to a JSON file where the data containers are properly translated to JavaScript Object Notation. It’s as simple as calling “json.dump(patientList, fileToWrite, indent=4)”.

The visual analysis tool also reads in *organAtlas.json*, using the listed organs to generate the organ filter list and populate the scenes with 3D models of the organs. The 3D organ models were extracted from a DICOM dataset using Slicer3D [13] and exported as vtk (XML format) models [20]. The set of organ models is standardized across all patients read in by the application.

Main View



Patient 5078
Total Dose: 70 GY GTV: 49.5 cc (R) BOT

Patient 5081 0.97781
Total Dose: 70 GY GTV: 41.1 cc (R) BOT

Patient 10159 0.97193
Total Dose: 70 GY GTV: 49.9 cc (R) Tonsil

Patient 215 0.96857
Total Dose: 70 GY GTV: 40 cc (R) BOT

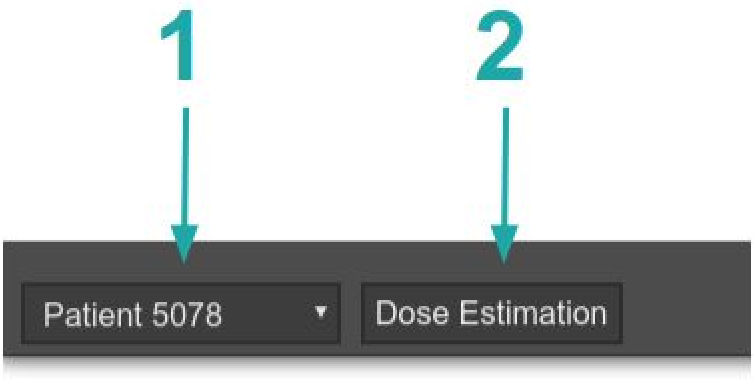
Patient 10044 0.95103
Total Dose: 70 GY GTV: 65.6 cc (R) Tonsil

Patient 183 0.94858
Total Dose: 70 GY GTV: 43.2 cc (R) BOT

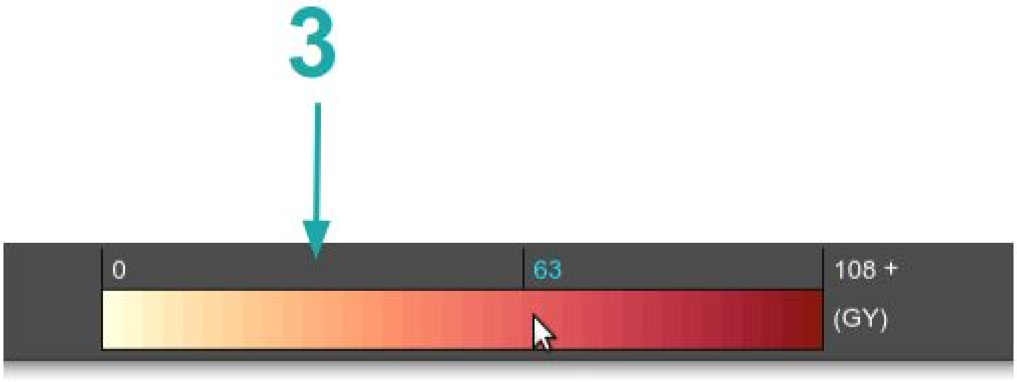
Patient 220 0.94625
Total Dose: 70 GY GTV: 34.7 cc (R) BOT

Patient 257 0.94307
Total Dose: 70 GY GTV: 38.3 cc (L) BOT

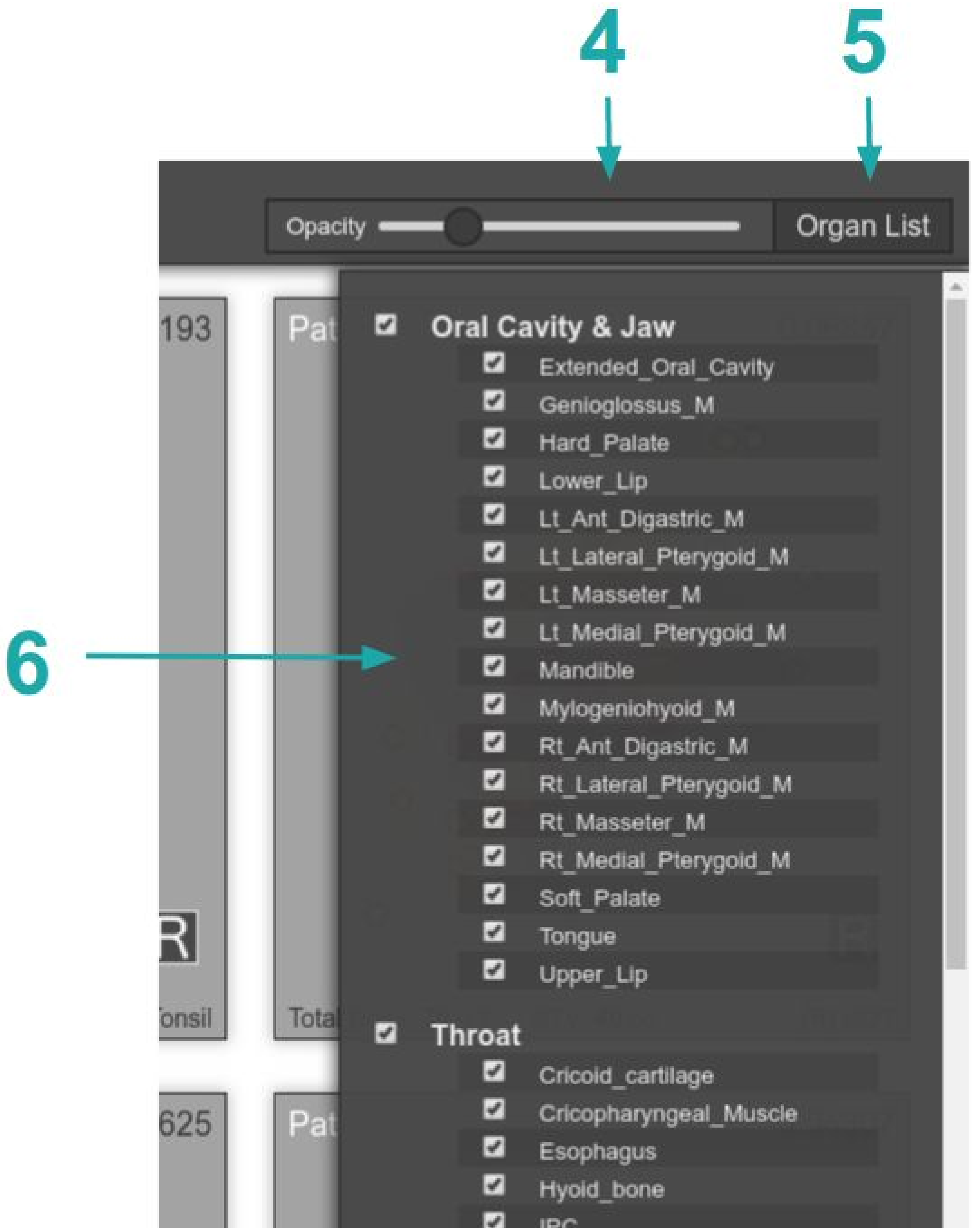
When the web-application starts, the user is greeted with the above view. A default patient is selected for analysis. Each block represents a patient, with organs of the head and neck encoded as spheres and 3D models. In the view above, the user is looking at the right side of each patient's head. The selected patient is located at the front of the list of patients, in the upper left corner. The rest of the patients are ordered by their similarity score compared to the selected patient. In this case, Patient 5081 is the most similar to Patient 5078, followed by Patient 10159, then Patient 215, etc. By default the top fifteen patients are shown. When you scroll to the end, there are links to either show or hide five more patients. It is possible to "show" all 101 patients and scroll through the entire list. The application handles fairly well because only the patients in the window's view are actually being rendered, optimizing overall performance.



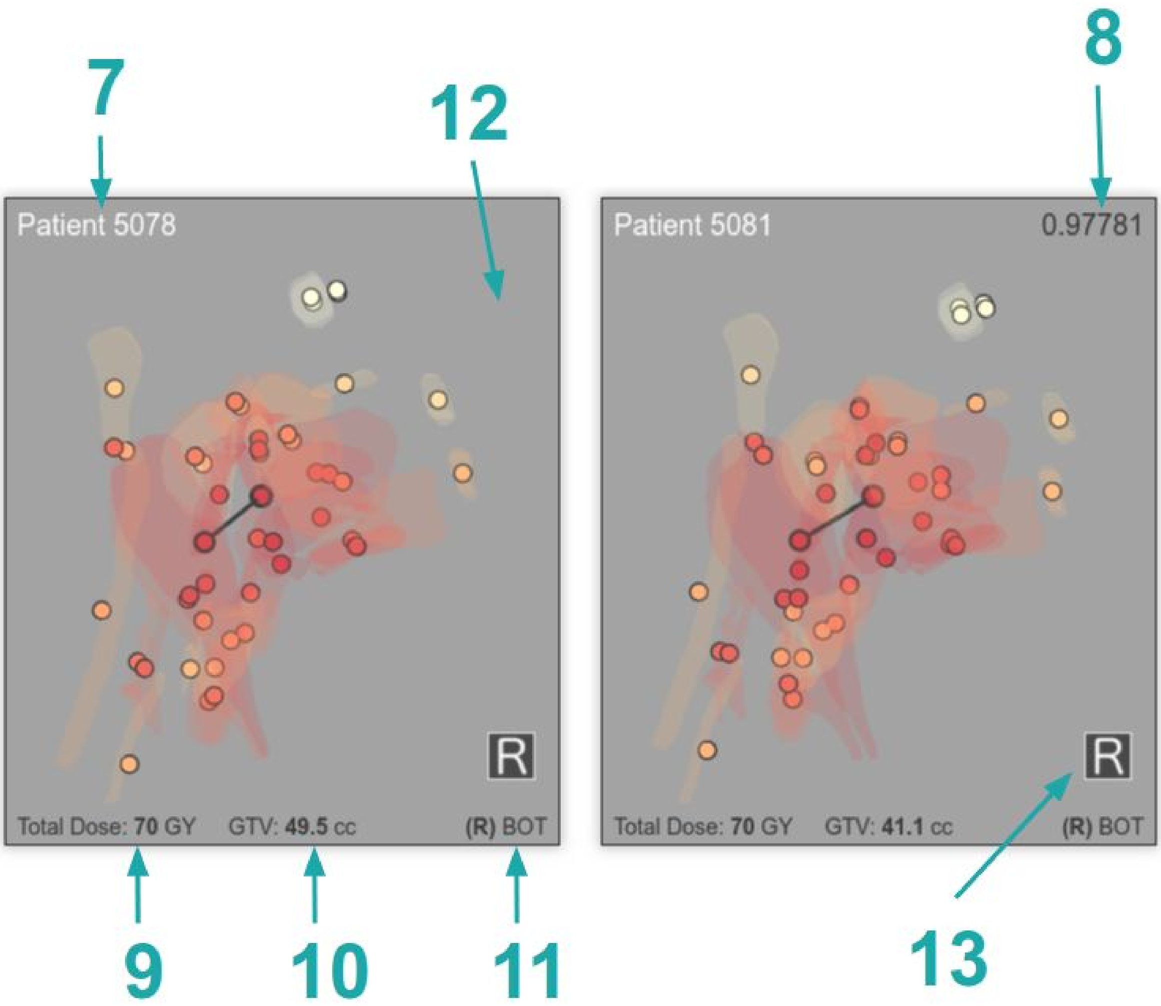
1) The user can select a target patient using the dropdown menu.
2) Pressing the button takes the user to the Dose Estimation view.



3) The color scale indicates the level of radiation a organ has received, measured in GY. Hovering over the color scale shows the dose value mapped to that color. *D3.js* is used to color encode the organs, mapping a range of radiation doses to the custom color scale.

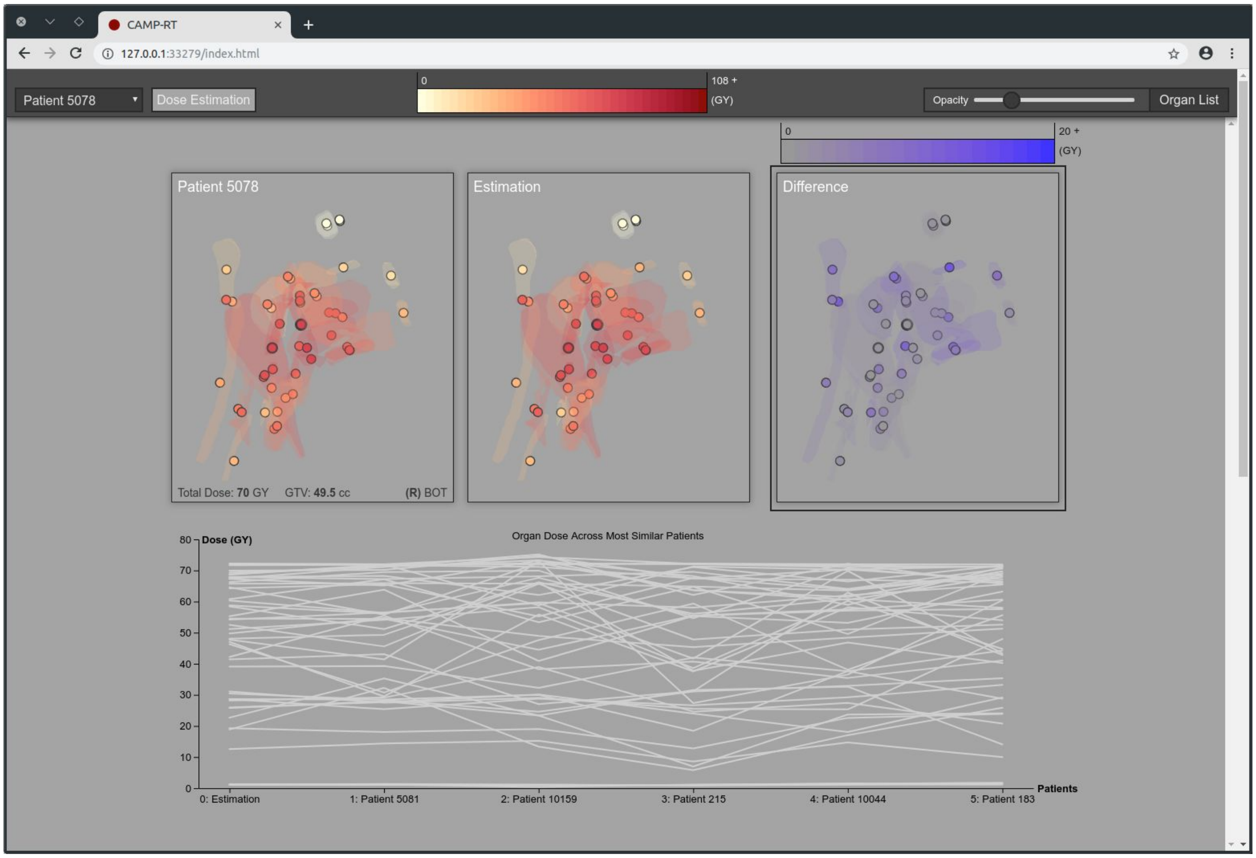


4) A slider used to change the opacity of the 3D organ models.
5) Toggles the Organ List menu.
6) The Organ List menu. The list is dynamically created at runtime using the data stored in *organAtlas.json*. The menu is used to either filter (hide) a single organ or an entire partition.

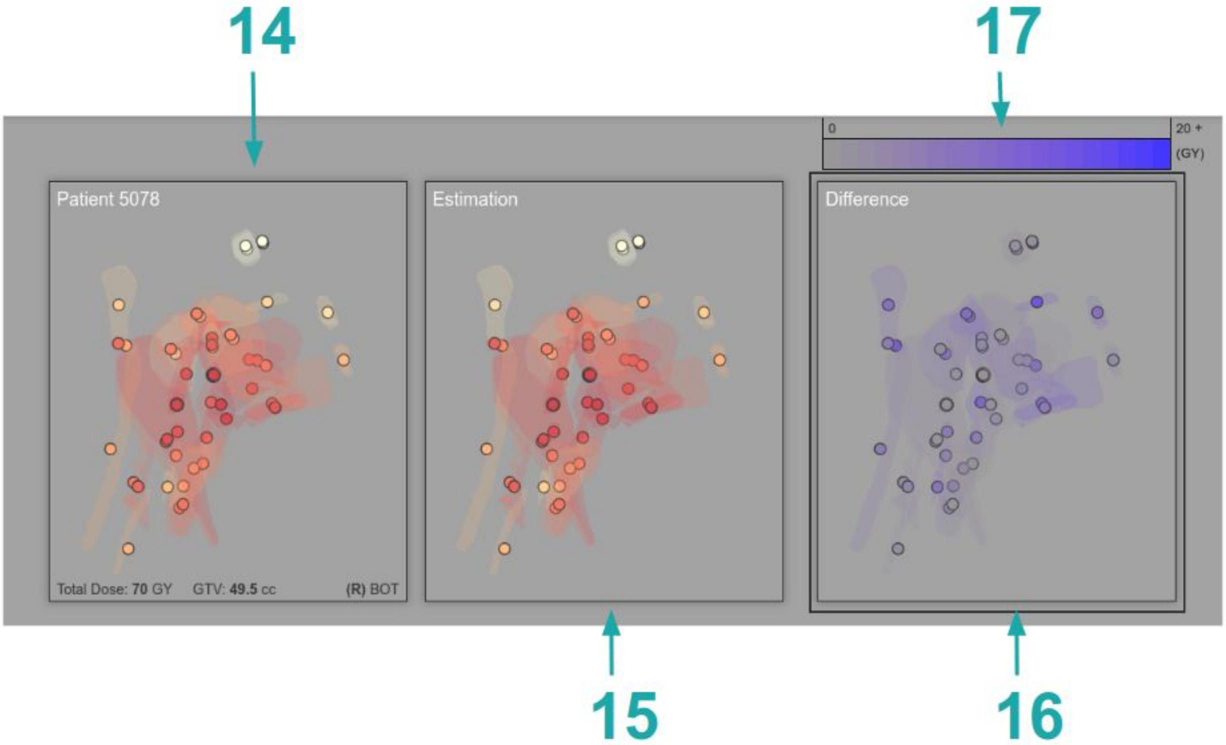


7) The patient's name or ID.
8) The patient's similarity score when compared to the target patient.
9) The total dose a patient received or will receive.
10) The volume of the primary GTVp tumor.
11) Tumor laterality and subsite.
12) A three.js 3D scene. Organs are represented two different ways, as 3D models and at the centroid of each model as a sphere. Adjusting the opacity slider, a hybrid organ view can be set. Spheres are used to minimize occlusion, allowing the user to see all data points and patterns clearly. Models of organs are used to help distinguish between different organs and to keep the overall structure of the head and neck intact. The opacity level of the models tend to depend on the user task at hand. A patient's organs are spatially mapped to the 3D scene, and each patient can be rotated to get a better look at the data. Tumors are only represented as spheres, because models can not be obtained. The spheres are slightly larger than the spheres representing organs, and the black outline is thicker. If a patient has both a primary GTVp tumor and secondary GTVn tumor, a solid black edge is placed between the two.
13) A orientation cube, to aid in determining what side of the head is being viewed. Currently the right side of the head is being viewed.

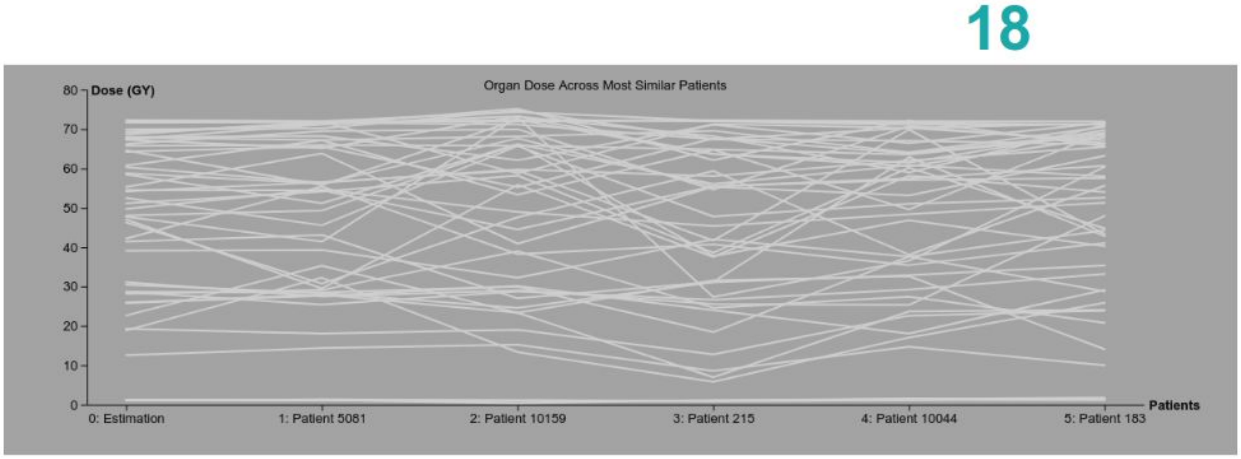
Dose Estimation View



In the Dose Estimation view, the top five most similar patients are averaged to provide a dose distribution prediction for the target patient. In this view, the dose prediction for a organ can be compared to the actual radiation dose applied to that organ. An additional patient view is created to highlight the differences between the predicted and actual values. Users can use this view to validate the current similarity method.

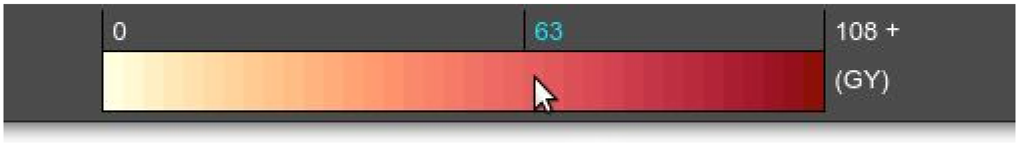


14) The target patient from the main view is also presented in the Dose Estimation view.
15) The top five most similar patients are averaged to create the Estimation view.
16) A third view of the selected patient, highlighting the differences between the predicted and actual doses.
17) The color scale used to amplify the differences, making them easier to observe.



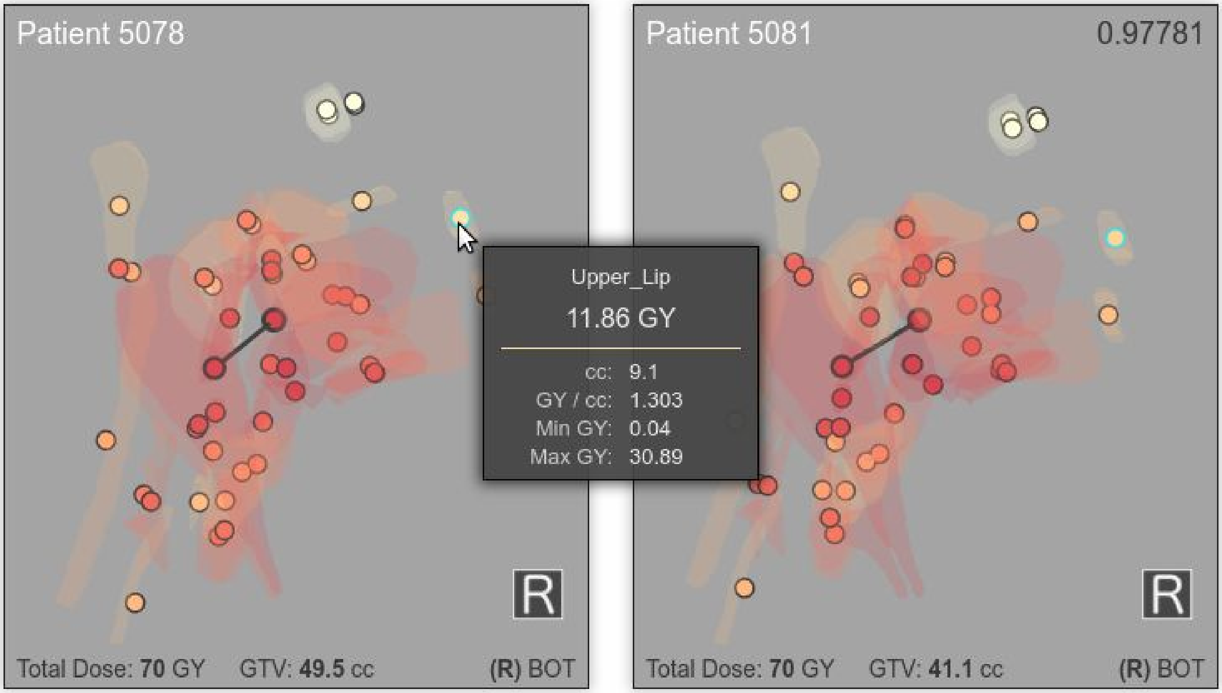
18) A multi line chart, showing the data that goes into the prediction. In the main application view, organs are represented as spheres and models. Here, each organ is represented as a line. Furthermore, in the main view the color of the sphere and model correlates to the radiation dose received. Here, the height of the line maps to the radiation dose received. The y axis plots the radiation dose. The x axis holds the estimation, followed by the top five most similar patients compared to the selected patient. The Estimation view of the patient in the top row corresponds to “0: Estimation” on the x axis. The chart can be used to observe trends or anomalies in the top five most similar patients.

Interaction



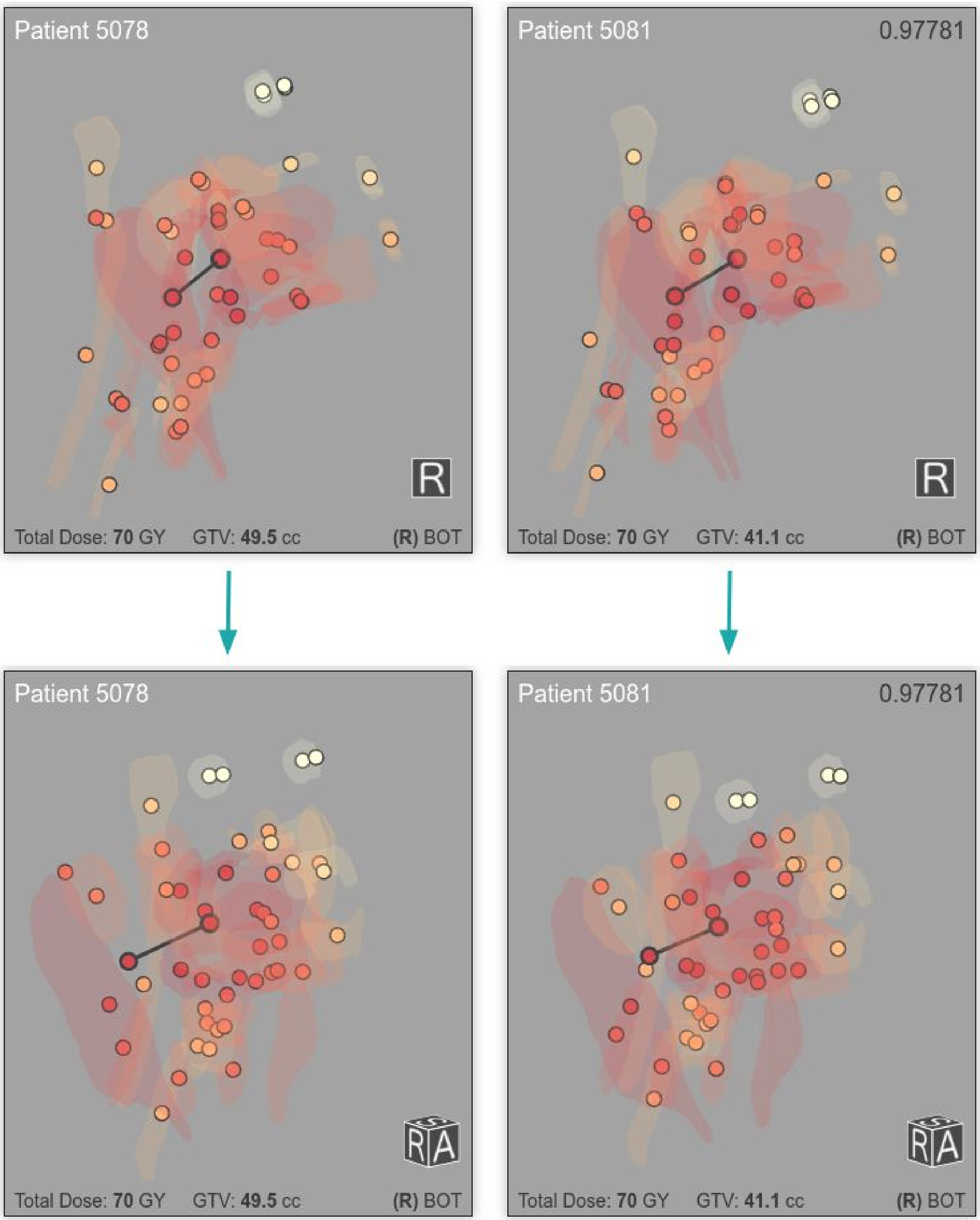
Details on demand.

As mentioned earlier, hovering over the color scale shows the dose value mapped to that particular color.



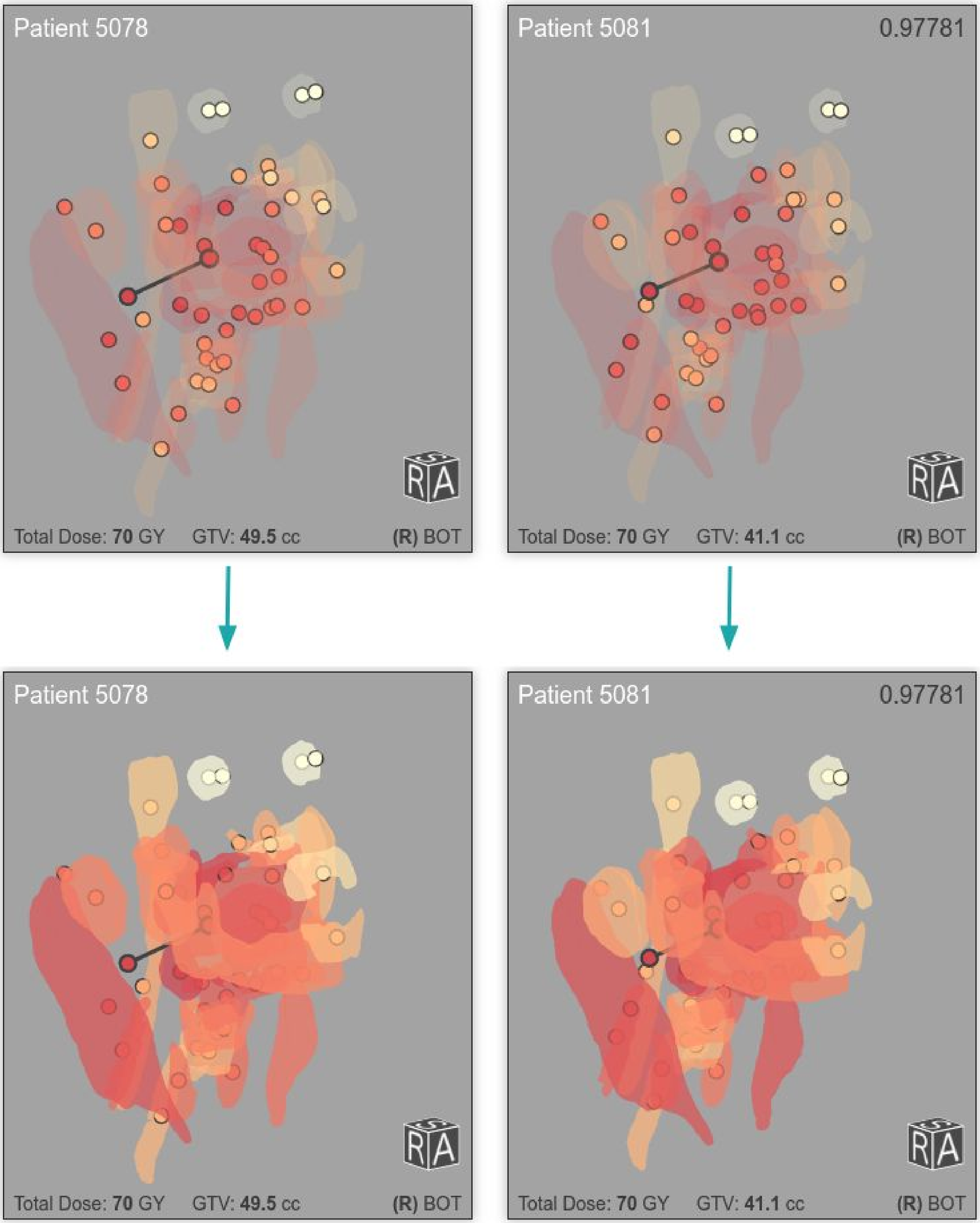
Linked Views plus details on demand.

Hovering over a sphere, details about the respective organ appear. Information shown includes the name of the organ, mean radiation dose received, volume, dose per volume, minimum radiation dose received, and maximum radiation dose received. The patient views are also linked. Hovering over a organ highlights that organ across all patients.



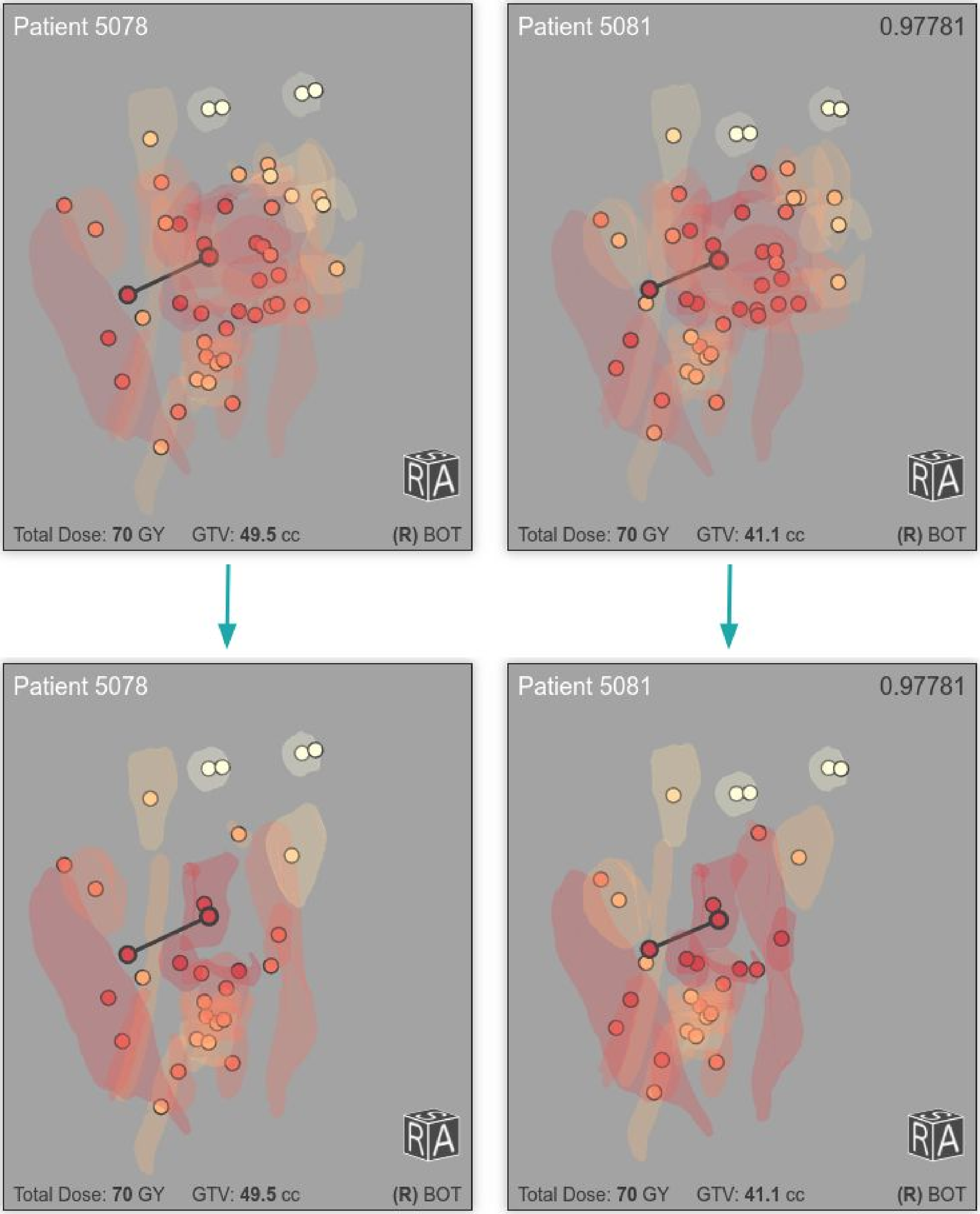
Linked Views. Rotation.

The patient views are further linked through rotation. Rotating one patient syncs the rotation across all patients. Consistent views of the patients ensure appropriate observations can be made.



Linked Views. Opacity.

Adjusting the opacity slider changes the opacity of the 3D organ models. The patient views are further linked through opacity. Decreasing the opacity makes it easier to understand the structure of the head and neck, and the individual organs. Increasing the opacity allows users to view the radiation doses across all organs without having to rotate the scene.



Patient 5078

Total Dose: **70** GY GTV: **49.5** cc **(R)** BOT

Patient 5081 0.97781

Total Dose: **70** GY GTV: **41.1** cc **(R)** BOT

Patient 5078

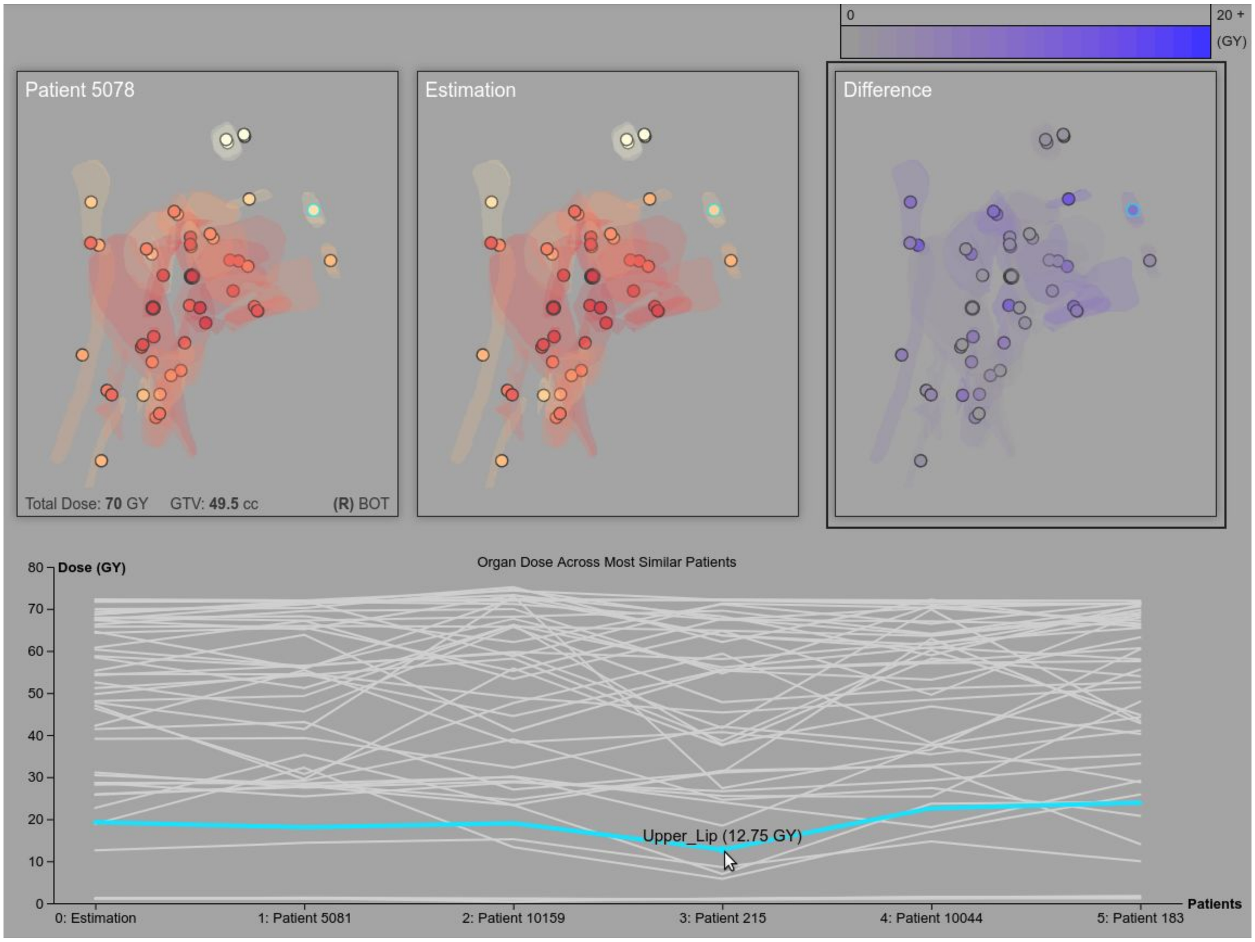
Total Dose: **70** GY GTV: **49.5** cc **(R)** BOT

Patient 5081 0.97781

Total Dose: **70** GY GTV: **41.1** cc **(R)** BOT

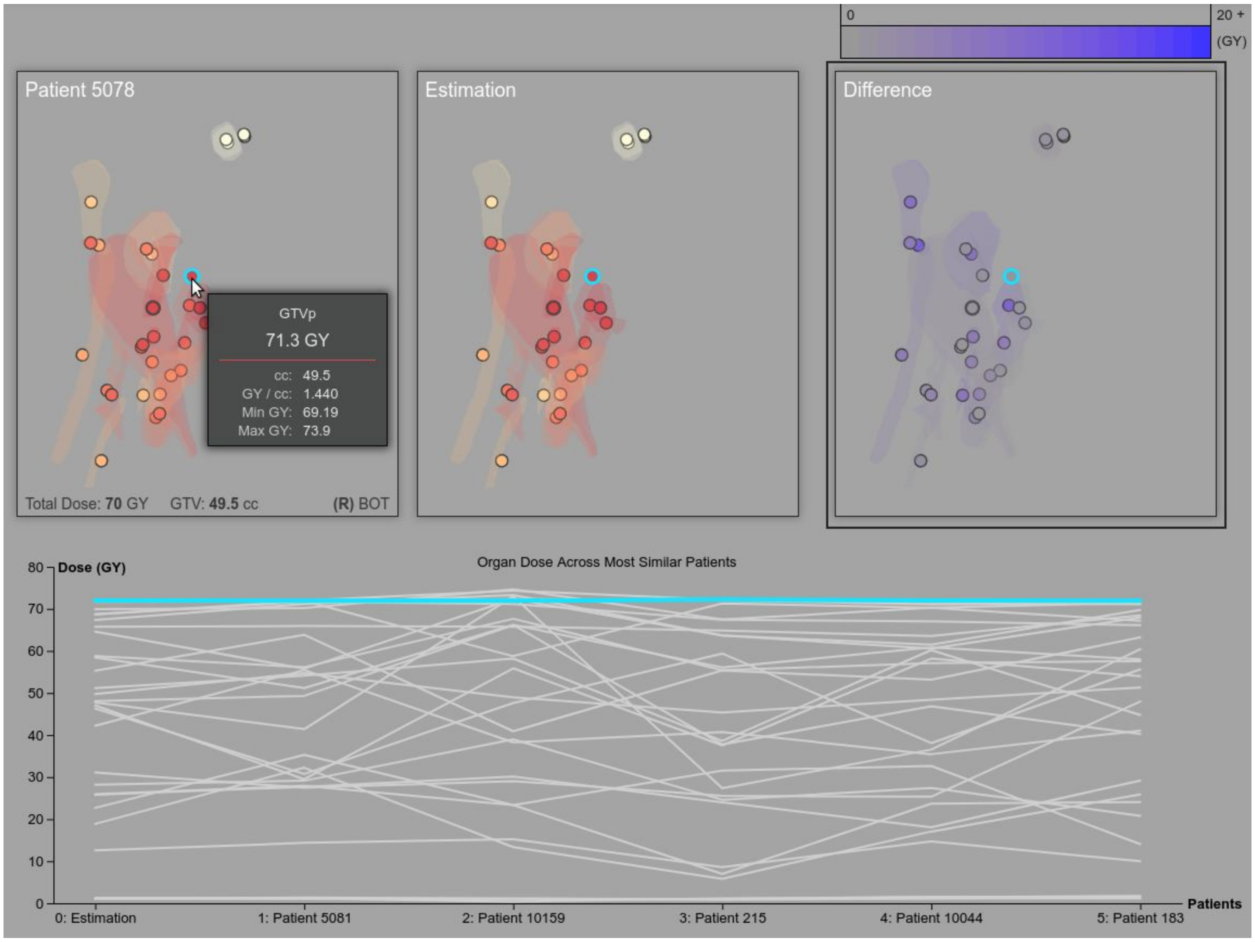
Linked Views. Filtering Organs.

It is possible to filter organs from the scene. Filtering of the organs is linked across all patients. Organs can be filtered individually or by partition. In the above, the Oral Cavity & Jaw partition is filtered out.



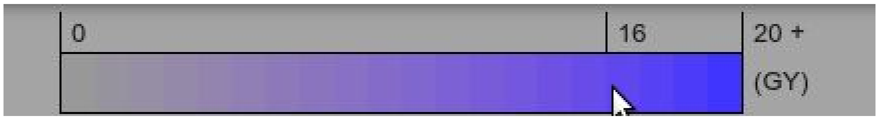
Linked Views across different encodings. Details on Demand.

In the Dose Estimation view, all the visual encodings are linked. Hovering over a line in the chart highlights the respective organ (sphere) across the top views. Hovering over a line also reveals the name of the organ, along with the radiation dose the organ received for the patient closest to the mouse position.



Linked Views across different encodings. Details on demand. Filtering organs. The thicker outline of the spheres representing tumors makes them easier to see when highlighted.

In the previous view the user was hovering over a line in the chart. In this view, the user is hovering over a sphere. Again, all the visual encodings are linked. Hovering over a sphere highlights the respective sphere across all the top views, and highlights the respective organ (line) in the multi line chart. Filtering of the organs is also linked across all encodings.



Details on demand.

Just like with the other color scale, hovering over this color scale shows the dose value mapped to that particular color.

Implementation

The visual analysis tool was developed with web technologies JavaScript [14][15], HTML5 [16], and CSS3 [16]. The web-application was tested using the Chrome v70 web browser [17]. Three.js r84 [18] is a JavaScript 3D library, and is used to spatially render and interact with each patient and organs of interest. D3.js v5.7 [19] is a JavaScript library for manipulating documents based on data. I use D3 to create and render a multi-line chart, map custom color scales to dose values, and read in JSON files.

3. EVALUATION AND RESULTS

I evaluate quantitatively the results of the prediction using a dataset of 101 head-and-neck cancer patients. I further report qualitative feedback from our collaborators regarding the method and interface.

3.1 Quantitative Evaluation

Our hypothesis is that the more similar two patients are under the developed similarity method and metric, the more likely it is that those patients will receive similar radiation therapy treatments. To test this hypothesis, we use an existing dataset of 101 head-and-neck cancer patients.

For each patient in this dataset, we first use the similarity method to identify the five most similar patients to the given patient. We then use the known radiation dosage across the five most similar patients to predict the radiation dosage distribution (the amount of radiation per each organ) of the given patient. We validate the prediction against the actual radiation dosage distribution for that patient.

As discussed above in the Dose Prediction section, the Python script writes to disk the file *differences.csv*. The columns consist of the patient's ID, followed by the IDs of the top five most similar patients, and then every organ difference. Two additional columns are created at the end, "Sum" and "Average". The "Sum" column calculates the sum of all the organ differences for a particular patient. The "Average" column calculates the average organ difference for a particular patient. The *differences.csv* file is used to evaluate quantitatively the results of the predictions and for k-fold cross-validation. The goal of cross-validating the data is to test the method's ability to predict new data that was not used in estimating it. We are in the very early stages of applying k-fold cross-validation. Below are some findings from our preliminary evaluation, which were presented to our collaborators.

Looking at the “Average” column, 24 patients have an average difference dose greater than or equal to 8 GY, while 77 patients fall below this value. We examined the 24 patients using the visual analysis tool, gathering any insights that might explain the high averages. Some recurring themes were observed. 9 patients do not have a primary GTVp tumor, or the data is missing. This means proper similarity scores could not be computed which could account for the high averages. For 10 patients, either the target patient or a patient used for the prediction is a special case where radiation is only applied to one side of the head, skewing the final dose distribution prediction. The current similarity algorithm can't detect these special cases, and development is underway to achieve this functionality. 4 patients are above the 8 GY value for reasons currently not known, but for these patients the organs in the throat partition are either highly overestimated or highly underestimated when compared to the actual radiation dose values. This would be interesting to explore further. For 1 patient, the similarity scores for the top five most similar patients are relatively low, which can help explain why the average difference dose is larger.

3.2 Domain Expert Qualitative Feedback

To evaluate our interface, we have collected qualitative feedback from four radiation oncologists at MD Anderson, and have demonstrated the interface to larger groups of oncologists at the UIC Cancer Center, at the John Theurer Cancer Center, and at a computational pathology meeting in Shonan, Japan. The feedback was uniformly enthusiastic. Domain experts have repeatedly commented on how intuitive the 3D views were to understand the radiation dosage distribution similarity across patients. During repeated evaluation sessions, our collaborators were furthermore able to identify outliers in the data and a few instances of defective data, which further testifies to the value of this tool.

4. DISCUSSION AND CONCLUSION

The automated method and visual analysis tool have undergone many iterations, which correlated closely to the type and amount of data provided by our collaborators. There was once a time where we were manually extracting data components from DICOM files, before the workflow across collaborators was streamlined. As another example, the first iteration of the visual analysis tool only contained 2D views of the patients, because the exact locations of organs were not known and each patient only had around ten organs associated with them. When we received more data, and each patient had around fifty organs associated with them, the visual encodings had to be updated to allow for effective visual analysis.

The current iteration can be thought of as a Beta release. We are getting consistent and promising results, but this is an ongoing research project that will be taken further.

ACKNOWLEDGMENTS

I thank my advisor, and our collaborators at MD Anderson Cancer Center, at University of Iowa, and at University of Minnesota. I thank Timothy Luciani for his help with the similarity method and algorithm, and the entire Electronic Visualization Laboratory for their help and support. I thank Andrew Johnson for reading and commenting on this report. CAMP-RT is part of the larger E-Radiomics in Precision Medicine project. This research is supported by NIH R01 NCI-R01CA2251.

REFERENCES

1. Medical Imaging & Technology Alliance, DICOM Standard PS3.1 2018e, 2018. https://www.dicomstandard.org/current/

2. Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, Apr. 2004.

3. Matlab implementation of SSIM method https://github.com/josejuansanchez/ssim

4. ECMA International,The JSON Data Interchange Format 2nd Edition, France, 2017. https://www.ecma-international.org/publications/standards/Ecma-404.htm

5. G. van Rossum and F.L. Drake (eds), Python Reference Manual, PythonLabs, Virginia, USA, 2001. Available at http://www.python.org

6. Numpy v1.15.4 https://github.com/numpy/numpy

7. MATLAB Runtime https://www.mathworks.com/products/compiler/matlab-runtime.html

8. MATLAB Arrays in Python https://www.mathworks.com/help/compiler_sdk/python/use-matlab-arrays-in-python.html

9. Python “glob” Module https://docs.python.org/2/library/glob.html

10. Python “csv” Module https://docs.python.org/2/library/csv.html

11. Python “json” Module https://docs.python.org/2/library/json.html

12. differences.csv https://docs.google.com/spreadsheets/d/1sJ6e-6i7owP6hos5PpvUbccqrA6P6X_nTI9kRsf2jAM/edit?usp=sharing

13. Fedorov A., Beichel R., Kalpathy-Cramer J., Finet J., Fillion-Robin J-C., Pujol S., Bauer C., Jennings D., Fennessy F.M., Sonka M., Buatti J., Aylward S.R., Miller J.V., Pieper S., Kikinis R. 3D Slicer as an Image Computing Platform for the Quantitative Imaging Network. Magn Reson Imaging. 2012 Nov;30(9):1323-41. PMID: 22770690. PMCID: PMC3466397. https://www.slicer.org/

14. ECMA International, ECMAScript® 2018 Language Specification 9th Edition, France, 2018. https://www.ecma-international.org/publications/standards/Ecma-262.htm

15. Brendan Eich, JavaScript, 1995. https://www.javascript.com/

16. W3C, HTML & CSS, USA, 2016. https://www.w3.org/standards/webdesign/htmlcss

17. Google Chrome Web Browser https://www.google.com/chrome/browser/

18. Ricardo Cabello, Three.js r84 JS Library, 2017. https://threejs.org/

19. Mike Bostock, D3.js r5.7 JS Library, 2018. https://d3js.org/

20. Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), The Visualization Toolkit (4th ed.), Kitware, ISBN 978-1-930934-19-1