VIBE: Visualization of Intrafraction Behavior from Electromagnetic Tracking

CIS 499 SENIOR PROJECT WRITEUP

Tamar Nevo
Advisor: Dr. Norm Badler
University of Pennsylvania

INTRODUCTION

Collaborating with a group of Radiology Oncologists at the Hospital of the University of Pennsylvania (HUP), my project aims to visualize prostate motion during a single radiation treatment by using an organ-tracking device. The natural movement of organs in the body can interfere with the accuracy of treating target tissue. As such, there has been an effort in radiation therapy to improve the precision of cancer treatments, reducing side effects by decreasing unintentional doses reaching normal tissue and better controlling tumors. Recent electromagnetic tracking using a technology called the Calypso 4D Localization System has provided means of recording the location of the prostate in real-time during radiation therapy. The Calypso system relies on three Beacon® transponders, tiny electromagnetic sensors implanted in the prostate prior to treatment, which monitor the position and motion of the organ. The spatiotemporal data from the three transponders is recorded in an excel file in a sequential order (i.e. T1, T2, T3, T1, T2…), and must be re-sampled to explain the position and movement of each transponder independently over time (b-spline curves will be used for this task).

Presently this technology is being used for treating prostate cancer, but it is expected to be useful for radiation cancer treatments of localized tumors in any organ subject to involuntary motion, such as the breast or lungs (due to breathing).

Significance of Problem or Production/Development Need

Many side effects of the Calypso treatment are thought to have been caused by unnecessary radiation reaching "healthy tissue" due to organ movement during the treatment. The goals of the project are to import, analyze, and display the transponder data with respect to time in order to track the movement and deformation of the organ during the treatment. This aims to improve the accuracy of the radiation beam, thereby reducing the likelihood of damaging non-target tissue.

Technology

Calypso 4D Localization System
Beacon® Transponders
MATLAB
Excel (VBA)
Language used for interface (maybe web-based)

BACKGROUND

In attempt to improve the accuracy of radiation treatments on tumor-bearing tissue (specifically, the prostate) and to reduce doses on normal tissue, much emphasis has been placed on monitoring the location and deformation of the organ *during* therapy. Electromagnetic tracking has been used to follow the real-time motion of the prostate (e.g. at the Department of Radiation Oncology at the University of Florida and the University of California among others) with the help of the Calypso 4D Localization System. Beacon transponders were implanted in the organ and tracked for a period of time (usually between 5-20 min) to provide information about the fraction of time that the prostate was displaced by “x number of mm” for each session and patient. Transponder location over time has been plotted and analyzed. Visualization of the organ’s movement and deformation, however, has not been so widely explored.

ALGORITHM DETAILS

Phase 1 – Re-sampling Data

1. The first step is to convert all time strings to a single float number using a simple VBA parsing function. Another method I learned is by selecting the cells that contain the time strings and performing a “text to columns” built-in function under “Data.” It allows you to create separate columns around specific markers. I will go into this more later.
2. Because the Calypso machine records and outputs the location of each transponder sequentially, it is necessary to resample the data so that the position of all three transponders can be retrieved at any time. For this I wrote a function (in VBA) that creates three separate sheets in the original excel file that contain the x,y,z data for every recording of transponder 1, 2, and 3 respectively. A simpler method is selecting all four columns and doing “Data→Filter.” Using the created drop-downs, filter the first column to only show “1s” for example, then “2s” and “3s.”

3. I remove spurious data (zeros) by sorting either the x, y, or z column in ascending order to reveal the “zero” rows. Delete these rows and re-sort the data based on the ascending time column.
4. In Matlab, I wrote a function to graph each of the 9 arrays (T1x, T1y, T1z, T2x…), drawing a b-spline curve for each.
5. Using one consistent time sequence for each of the nine, I interpolate each b-spline to find values for target data times (essentially obtaining a triangle position for every second of the 18-minute procedure).
6. I place these values into nine columns in excel, and because the machine has a resolution of .1mm, I format the columns to display only one decimal place.

Phase 2 – Triangle Computations

1. For each target data time (every second = frame), the three transponders form a triangle. I begin by writing function to compute for each triangle the:
    a. Centroid
    b. Min/Max for x, y, z (the bounding box)
    c. Normal vector (unit length)
    d. Rotation axis and angle between first triangle normal and current triangle normal
2. With three points in space and min/max values, I can plot the triangle and its bounding box in 3D – the boundingBox() function generates this still image.
3. Following, I create an .avi file made up of a sequence of frames for every second of the 18-minute procedure – a “jiggling triangle” animation.

Phase 3 – 3D Contour Model of Organ

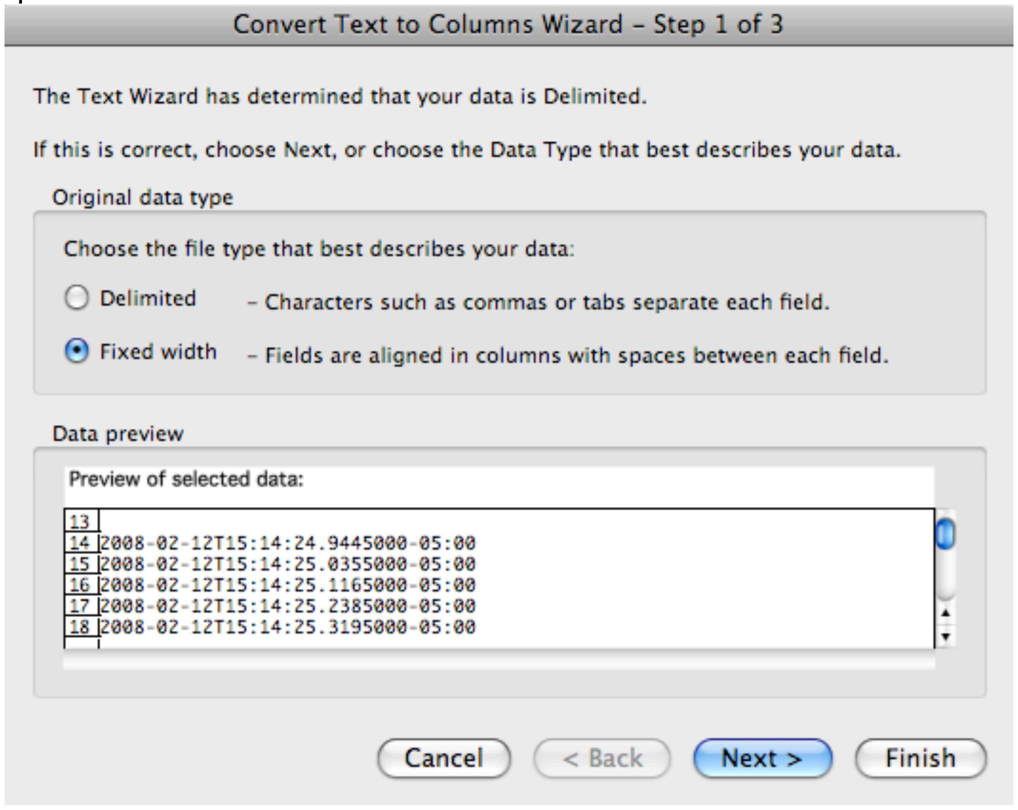
1. I first plot graphs of 1) Centroid movement in the X, Y, and Z directions, 2) The rotation angle between each frame and the first, and 3) the scale change in each direction.
2. A second supplied excel document provides a long list of points which define the contour of the prostate. Blank rows separate between contour loops. Before animating, I test the supplied data by displaying a still frame of the organ in MATLAB using a function I wrote called drawContours().
3. In order to animate, I compute the translation, rotation, and deformation of each triangle compared to the first triangle.
4. This information can then be applied to the contour data in sequence to create an animation, which is what is generated by calling animateContours() with arrays for the x, y, z positions of each contour point and the three points of the triangle

STEP-BY-STEP INSTRUCTIONS FOR RUNNING CODE

1. Open excel file with Calypso output. The data should look like this:

Transponder	X	Y	Z	PositionTimeString
1	0.768007685	-0.670575507	1.237602205	2008-02-12T15:14:24.9445000-05:00
2	0.543598218	0.305644489	-1.031243917	2008-02-12T15:14:25.0355000-05:00
3	-1.595396957	0.522831979	-0.420960678	2008-02-12T15:14:25.1165000-05:00
1	0.773501245	-0.685332166	1.253303285	2008-02-12T15:14:25.2385000-05:00
2	0.530271027	0.278862085	-1.017448201	2008-02-12T15:14:25.3195000-05:00
3	-1.606955031	0.507740164	-0.432733083	2008-02-12T15:14:25.4005000-05:00

2. First convert lengthy time strings to seconds:
    a. Highlight the cells with time string data in them (Note that though the long string spans several columns, it is actually only stored in the first, so highlight the cells in that column that have data – i.e. not text or blanks)
    b. It is important to highlight only the cells with data as opposed to selecting the entire column – this can be done easily by clicking on the first cell, scrolling down to the bottom and Shift+clicking the last cell
    c. From the top menu bar, select Data → Text to Columns… which will pull up a screen like this:


Convert Text to Columns Wizard – Step 1 of 3

The Text Wizard has determined that your data is Delimited.

If this is correct, choose Next, or choose the Data Type that best describes your data.

Original data type

Choose the file type that best describes your data:

○ Delimited – Characters such as commas or tabs separate each field.

● Fixed width – Fields are aligned in columns with spaces between each field.

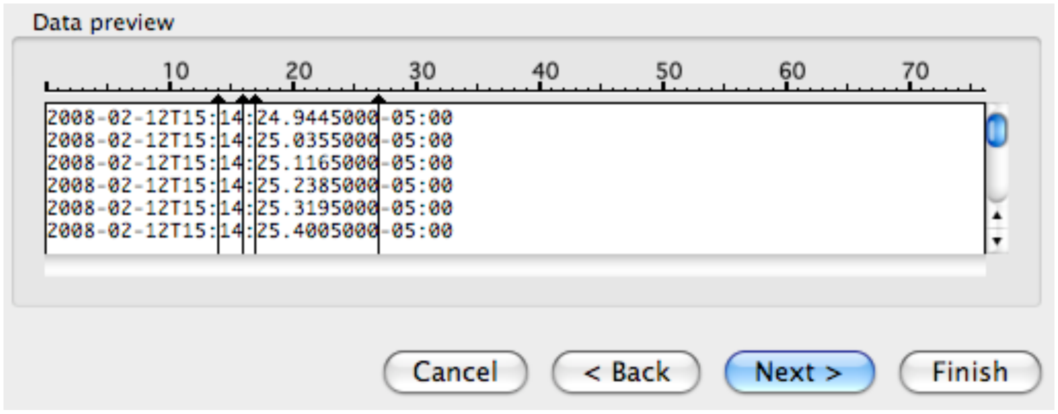
Data preview

Preview of selected data:

```
13
14 2008-02-12T15:14:24.9445000-05:00
15 2008-02-12T15:14:25.0355000-05:00
16 2008-02-12T15:14:25.1165000-05:00
17 2008-02-12T15:14:25.2385000-05:00
18 2008-02-12T15:14:25.3195000-05:00
```

    d. Choose “Fixed width” and press Next

e. Since we need to extract minutes and seconds, put dividers around these values like so:

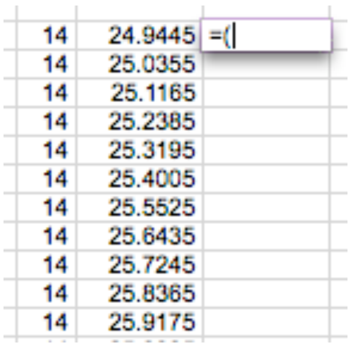


```
2008-02-12T15:14:24.9445000-05:00
2008-02-12T15:14:25.0355000-05:00
2008-02-12T15:14:25.1165000-05:00
2008-02-12T15:14:25.2385000-05:00
2008-02-12T15:14:25.3195000-05:00
2008-02-12T15:14:25.4005000-05:00
```

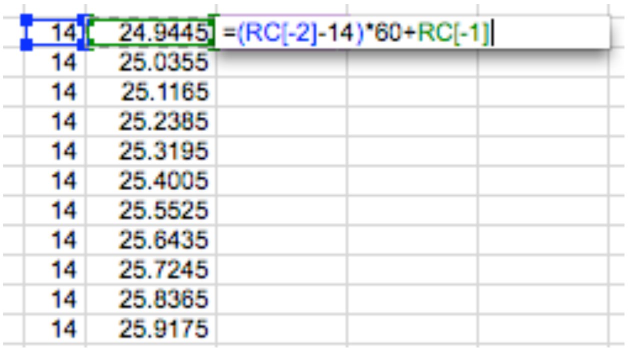
f. Press Next
g. Move your cursor over to the right to a blank column and select the cell in that column that is in the same row as the first time stamp entry – you are choosing where to print the new columns you've created
h. Press finish. Five columns should have printed
i. You can delete those columns that are not useful, keeping only the minutes column and the seconds column
j. In a new column, click on the cell that is inline with the first min/second entry. You will enter a formula here to compute seconds
k. To convert to seconds only, the minutes need to be multiplied by 60 and then seconds need to be added to this number – the first seconds entry will be used as the baseline. By this, I mean that if the first row has minutes = 14 and seconds = 24, then the formula would look like

(minutes-14)*60+seconds = 24.

l. In the cell that you've clicked on, start typing "=" then click on the minutes cell of that row and continue to type "-14*60+" if 14 is the starting minutes entry (otherwise replace 14 with the first minutes entry); then click on the seconds cell of that row; put parentheses around "(minutes-14)". It will look like this

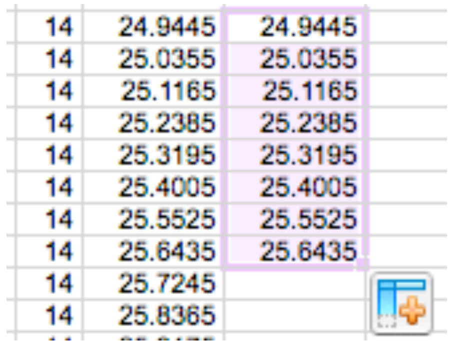


14	24.9445	=(
14	25.0355	
14	25.1165	
14	25.2385	
14	25.3195	
14	25.4005	
14	25.5525	
14	25.6435	
14	25.7245	
14	25.8365	
14	25.9175	



14	24.9445	=(RC[-2]-14)*60+RC[-1]
14	25.0355	
14	25.1165	
14	25.2385	
14	25.3195	
14	25.4005	
14	25.5525	
14	25.6435	
14	25.7245	
14	25.8365	
14	25.9175	

m. Press enter – the seconds will be computed. To apply to the entire column, click the lower-right corner of the computed cell and drag down for the entire column

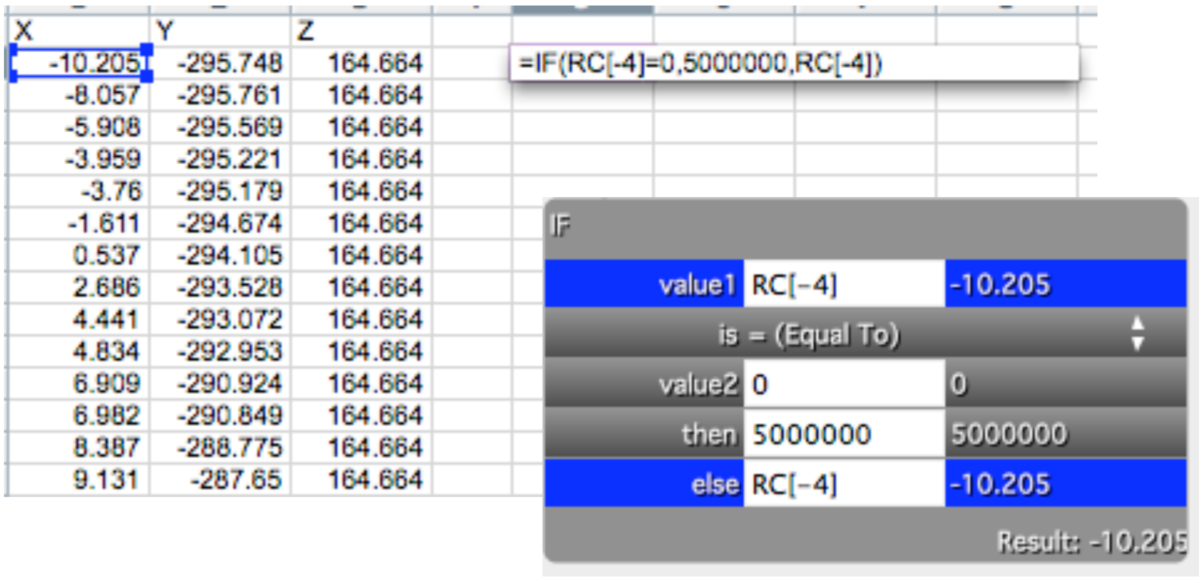


14	24.9445	24.9445
14	25.0355	25.0355
14	25.1165	25.1165
14	25.2385	25.2385
14	25.3195	25.3195
14	25.4005	25.4005
14	25.5525	25.5525
14	25.6435	25.6435
14	25.7245	
14	25.8365	

3. Next step: Isolate each transponder in a separate sheet:
   a. Highlight the 5 columns (by clicking on the letter at the top of the first column – say “C” for example – and dragging to include the other four, D E F G) These columns should now be a different color
   b. In the top menu, go to Data → Filter → Autofilter
   c. New drop downs should appear at the top of each column
   d. Click the first drop down (the transponder column) and select “1”
   e. This isolates all the transponder 1 recordings
   f. This is a good opportunity to remove all spurious data – click the drop down for any of the X, Y, or Z columns and select “Ascending.” All of the zero rows will move to the top of the list – delete these rows by highlighting the cells, right clicking → choose delete → shift cells up
   g. To get back the normal order, click on the drop down of the last column (timestrings) and select “Ascending”
   h. You can now highlight all five columns, copy and paste into a new sheet
   i. Follow steps d-g for “2” and “3”
4. Open the other excel file which contains the contour data – 3 columns, X, Y, and Z where each row represents a point in the contour
5. Because MATLAB does not recognize blank rows, but these rows are significant as markers of the “end of a contour loop,” we must replace blank rows with an arbitrary indicator value of “5000000”
   a. In a new column, click on the cell inline with the first contour point entry
   b. Under the top menu, select “Insert→Function…”
   c. A pop-up should appear called “Formula Builder”
   d. Double click on “IF”
      i. With the cursor blinking in the box for “value1” click on the cell that holds the first point’s X position
      ii. Then for the drop-down option select “=(Equal to)”
      iii. For “value2” enter “0”
      iv. For “then” enter “5000000”
      v. With the cursor blinking in the last box for “else” click again on the cell that holds the first point’s X position

    vi. This tells excel that if that cell is equal to 0, in this new column make it equal to 5000000, and if it is not equal to 0, keep it as is

e. It should look something like this



X	Y	Z
-10.205	-295.748	164.664
-8.057	-295.761	164.664
-5.908	-295.569	164.664
-3.959	-295.221	164.664
-3.76	-295.179	164.664
-1.611	-294.674	164.664
0.537	-294.105	164.664
2.686	-293.528	164.664
4.441	-293.072	164.664
4.834	-292.953	164.664
6.909	-290.924	164.664
6.982	-290.849	164.664
8.387	-288.775	164.664
9.131	-287.65	164.664

```
=IF(RC[-4]=0,5000000,RC[-4])
```

IF

value1	RC[-4]	-10.205
is = (Equal To)		
value2	0	0
then	5000000	5000000
else	RC[-4]	-10.205

Result: -10.205

f. Press enter and drag the bottom-right corner of the cell across (3 columns) and down (however many rows there are)

g. The three new columns should look identical to the original three except that the blank rows are replaced with "5000000, 5000000, 5000000"

h. Add one more entry to the end of each column with the value "5000000" to indicate the end of the final contour loop

6. In MATLAB…
    a. Creating arrays x, y, and z to generate contour object:
        i. In the contour excel document, highlight the new "x" column that you created (make sure it is all data and no text); copy this
        ii. In the MATLAB command window, type "x = [ ];" and in between the brackets, paste the new "x" column; then press enter
        iii. This creates an *array* called "x"
        iv. Do the same for y and z
        v. Once the x, y, and z arrays are created, type into the command window:
        contour = drawContours(x,y,z);

    b. Creating arrays p1, p2, and p3:
        i. In the Calypso output excel document, go to the new Transponder 1 sheet you created and highlight the X-column (again, make sure only to include data and not text cells), copy this
        ii. In the MATLAB command window, type "T1X = [ ];" and in between the brackets, paste the transponder 1 x-column; press enter
        iii. This creates an *array* called T1X

iv. Create a T1Y array and T1Z array in the same fashion

v. Also create a "T1times" array, inserting the new seconds column created earlier

vi. To interpolate, we want to find the location of each transponder at every second, so we must create a "newTimes" array → this can be easily done in excel. Choose a blank column anywhere and type 25 in one cell and 26 in the cell below it (rounded from the first seconds entry – so if the first entry were 12.2224, type 12 and then 13). Highlight both cells and drag down from the bottom-right corner until you reach the number closest to the last seconds entry, creating a list of numbers incrementing by one

vii. Copy this column and create with it a "newTimes" array

viii. In MATLAB make sure the directory is set to the correct place by clicking on the button on the top right and choosing the location where all the MATLAB files are stored

ix. In the MATLAB command window, type

```
[x_interp, y_interp, z_interp] = drawCurves(T1X,T1Y,T1Z,T1times, newTimes);
```

x. In the bottom left corner of MATLAB, three new rows should appear for x_interp, y_interp, and z_interp

xi. Double click on x_interp. A column of numbers should appear. Click on the top of the column to highlight it all and copy. Paste this column into a new sheet in the excel file. Double click on y_interp and copy/paste that alongside the x_interp column, and same for z_interp

xii. Next to these three columns, create a fourth column that is all semi-colons by typing ";" into the top cell and dragging the bottom-right corner down → this is helpful later in creating a matrix in MATLAB

xiii. Go to the Transponder 2 sheet you created and follow the same steps as for Transponder 1, this time using "T2X" "T2Y" etc. (newTimes is the same – you do not need to re-enter it) and typing into MATLAB

```
[x_interp, y_interp, z_interp] = drawCurves(T2X,T2Y,T2Z,T2times, newTimes);
```

xiv. Do this for Transponder 3 as well. You should have 12 columns in the new sheet that look like so

0.7747	-0.6769	1.2475	;	0.5342	0.2931	-1.0226	;	-1.5954	0.5228	-0.421	;
0.7688	-0.697	1.2589	;	0.5333	0.2774	-1.0197	;	-1.607	0.5077	-0.4327	;
0.7678	-0.7038	1.2608	;	0.5429	0.2716	-1.0313	;	-1.6188	0.5052	-0.4151	;
0.7735	-0.7056	1.2516	;	0.535	0.2717	-1.0317	;	-1.6022	0.4921	-0.4208	;
0.7647	-0.6966	1.2586	;	0.5358	0.2941	-1.0339	;	-1.6083	0.5133	-0.426	;
0.7601	-0.6822	1.255	;	0.5441	0.3006	-1.0331	;	-1.6076	0.5185	-0.4283	;

xv. Highlight them all, right click, select “Format Cells,” choose “Number” in the left column and change decimal places to 1. Click ok.
xvi. The final step in creating the p1, p2, and p3 arrays is to highlight the first four columns, copy
xvii. In MATLAB type “p1 = [ ];” and paste between the brackets. Press enter.
xviii. Highlight the next four columns, copy and paste into “p2 = [ ]”
xix. Highlight the last four columns, copy and paste into “p3 = [ ]”

c. FINALLY, in the command window type

```
animateContour_centered(contour,p1,p2,p3);
```

d. A new quicktime file called “contour_animation.avi“ should appear in the folder with the rest of the MATLAB files

REFERENCES

"Multi-Institutional Clinical Experience with the Calypso System in Localization and Continuous, Real-Time Monitoring of the Prostate Gland During External Radiotherapy." Int. J. Radiation Oncology Biol. Phys. Vol. 67, No. 4, pp. 1088–1098, 2007 Elsevier Inc.

"Observations on Real-Time Prostate Gland Motion Using Electromagnetic Tracking." Int. J. Radiation Oncology Biol. Phys. Vol. 71, No. 4, pp. 1084–1090, 2008 Elsevier Inc.

"Prediction of Intrafraction Prostate Motion: Accuracy of Pre- and Post-Treatment Imaging and Intermittent Imaging." Int. J. Radiation Oncology Biol. Phys. Vol. -, No. -, pp. 1–7, 2008 Elsevier Inc.

"Target Localization and Real-Time Tracking Using the Calypso 4D Localization System in Patients with Localized Prostate Cancer." Int. J. Radiation Oncology Biol. Phys. Vol. 65, No. 2, pp. 528–534, 2006 Elsevier Inc.

Verellen, Dirk, Mark De Ridder, Nadine Linthout, Koen Tournel, Guy Soete and Guy Storme. "Innovations in image-guided radiotherapy." Nature Review Cancer. December 2007, Nature Publishing Group, volume 7, 949-960.