MIXED REALITY
CAMERA CALIBRATION TOOL

INSTRUCTION MANUAL

January 2004
http://mixedreality.nus.edu.sg

Any commercial applications may be considered for separate licensing. In this case please contact

Industry & Technology Relations Office
National University of Singapore
10 Kent Ridge Crescent
Singapore 119260
Tel: 6874 2987
Fax: 6777 6990

Email: intquery@nus.edu.sg

The GNU General Public License (GPL)

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

QUICKSTART:

1. Print out the chequerboard pattern in the accompanying .pdf file and attach to a flat card surface.
2. Start the Calibration Tool Program
3. Accept the default settings in the dialog box that appears on startup
4. Select the camera source – for most users this will be "windows video stream" – you should now see the live video feed from your camera
5. Move the pattern back and forth in front of the video stream – try to vary the angle and position of the camera. If this process is successful you will see three colored bars move from the left to the right of the main window. These represent the quality of the captured data set (see Figure).
6. When all three bars are more than 2/3's of the way across the window, click on the "Calibrate" icon (CAL) on the toolbar to bring up the calibration dialog
7. Press the "Calibrate" button and wait. After a few moments, the calculated parameters will be displayed.
8. Save the parameters to file using the "File/Save" menu item.

INTRODUCTION

Need for camera calibration

All computer vision applications which need to make absolute measurements of the world need camera calibration. In short, this process measures the properties of the camera, and compensates for any distortions introduced by the lens. Some of these properties are familiar. For example, the horizontal field of view of the camera. Others are essentially abstract mathematical quantities which describe the imaging process. After calibration, we are able to establish which positions in the real world project to which pixels in the image.

Camera calibration is particularly important for mixed/ augmented reality applications. In addition to the need to make metric measurements of the world, we must also successfully align virtual objects with the video stream. In order to do this, we must simulate the parameters of the real-camera projection process (field of view etc.) when we draw the virtual objects. If the calibration parameters are incorrect, this superposition will be inaccurate (see Figure 1). Moreover, the if lens distortion is not corrected for the image will appear warped – for example, the edges of the chessboard in Figure 2 appear curved although they are really straight.

The aim of this program is simply to measure 9 parameters describing the camera projection model and save them to a file. This can be loaded by any computer vision application that uses that particular camera. The calibration process involves presenting a known "chess board" type pattern to the camera. Although such techniques have been available for some time, we hope to have rendered the process as fast and painless as possible with this utility.

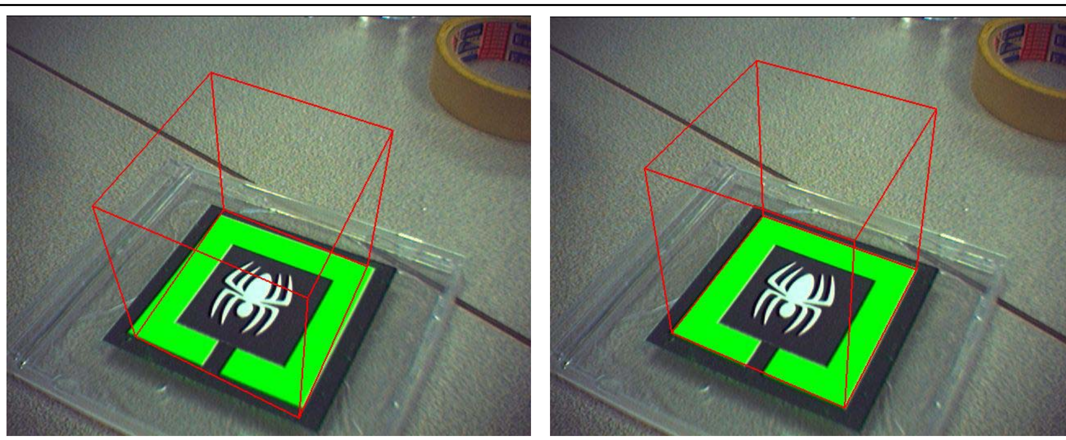


Figure 1. The importance of calibration is demonstrated in these pictures. The left shows the superposition of a cube on the marker, with some "sensible" assumed parameters. However, the cube does not successfully line up with the edge of the marker. The right shows the superposition with the correct parameters. The cube now sits exactly along the edges of the marker.

INSTRUCTIONS

Calibration Patterns and Set Up

In order to calibrate the camera, the user is required to move a calibration "chessboard" pattern in front of the camera. The computer stores these frames and subsequently uses them in order to calculate the parameters. A sample pdf of a calibration target is provided in the "calibration" directory. The user should print this out and mount it to a smooth, flat surface.

When the calibration utility is started up, a dialog box appears and the user needs to input the number of squares in the x and y directions (see figure 2), as well as the size of the squares in mm and the color of the 'u' shaped border – the default is black, but the program can also recognize bright green borders. The default parameters are correct for the pdf provided, but the advanced user may want to generate his or her own grid patterns. If this is the case, then please take pains to set these parameters correctly - if these settings are incorrect, the resulting calibration procedure will not work and the program must be restarted.

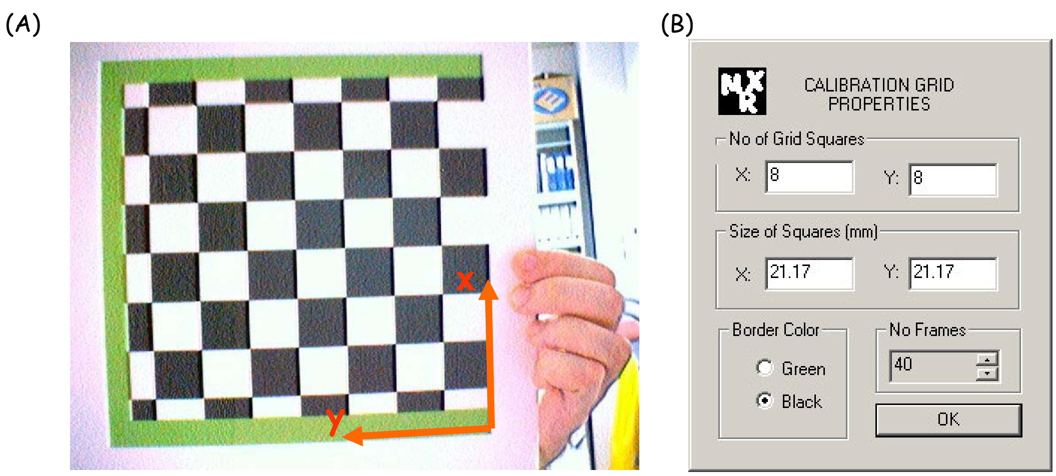


Figure 2. (A) Shows a typical calibration pattern with an "u" shaped green border. The x and y directions of the chessboard relative to the "u" are shown. (B) shows the calibration grid settings dialog which appears on startup.

Camera Initialization

When the user quits the dialog box, the main application will start. In the main window, the user should see "tv static" type noise. This indicates that the user's camera is not currently initialized. To start the live camera stream, select the appropriate item from the "camera source" menu. For most users, "windows video stream" will be the correct option.

Capturing Frames

The Calibration utility has two main modes, which can be swapped between using the round green button on the toolbar (). In the capture mode (depressed), the live video stream is shown. The program analyses the video stream and saves suitable images where it is able to extract the positions of the points on the chessboard. When not in capture mode, the user can review these saved images.

Three bars are shown moving along the bottom of the screen (see Figure 3). The topmost, yellow bar shows the proportion of the total number of frames that have been captured. The middle and lower bars measure the quality of the data set captured so far. The middle bar is a measure of the variety of angles of the chessboard that have been captured. The lower bar is a measure of the area of the screen that has been covered by the chessboard across the whole data set. As more frames are captured, the program will intelligently replace frames it has already captured if they increase the overall quality assessments. The user should manipulate the chessboard pattern until they are satisfied with these measurements of the quality of the data set. A good rule of thumb is that both indicators should be more than 2/3 of the way across the window before calibration proceeds.

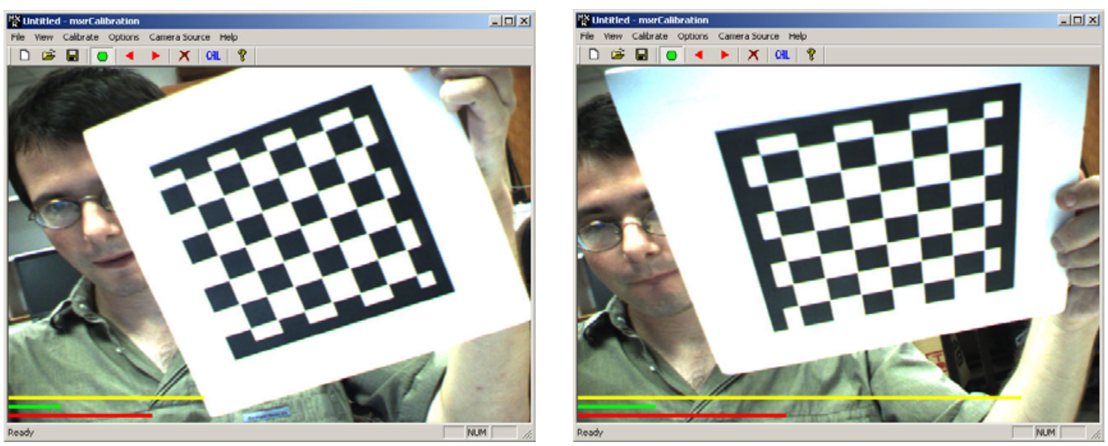
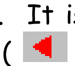
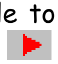


Figure 3 Capture of calibration target. For optimal results, the target should be manipulated so that it is presented at a variety of angles, and it covers all parts of the screen. Three colored bars at the bottom of the screen assess the quality of the captured data set.

Viewing Stored Frames

When the capture mode button () is clicked again, we enter review mode. It is now possible to review the frames that the program has captured. The previous () and next () frame buttons can be used to move through the data set. A counter at the bottom informs the user which frame he is currently viewing. Red dots are superimposed over the corners of the chessboard to indicate where the computer extracted points from the frame (see Figure 4). If the points do not appear to be accurately placed or if the frame is blurry or otherwise degraded, the frame can be abandoned by pressing the cancel button (). This irreversibly throws away that frame.

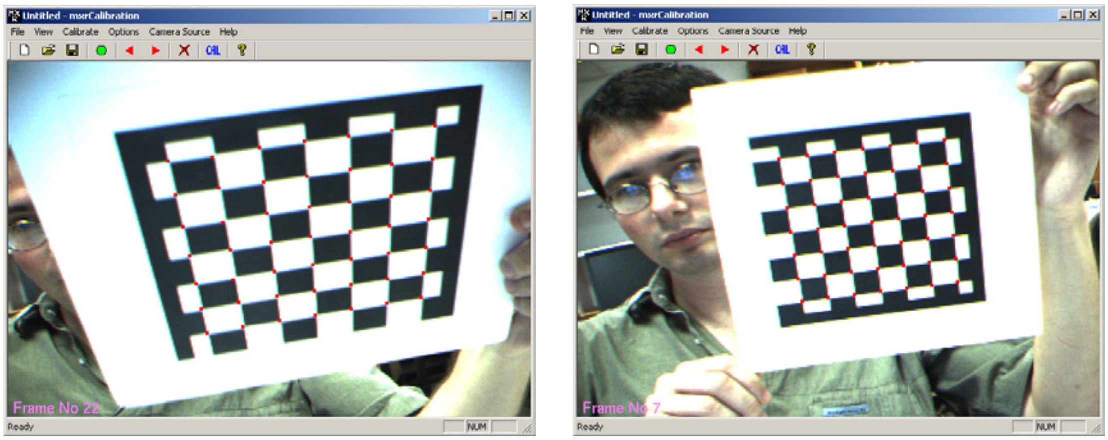


Figure 4. Review mode. The user can cycle through the stored frames and assess them. Frames where the quality is poor can be abandoned.

The user should use review mode to ensure that a wide and balanced variety of angles has been selected. If almost the whole data set consists of viewing the chessboard at one particular orientation, it is unlikely to provide a good substrate for the subsequent calibration computation.

Calibration

In order to calibrate the camera, press the calibrate button on the toolbar (). This will bring up the main calibration dialog box (see Figure 5). This displays the current calibration parameters. If the user has collected insufficient data, the calibration button will be inactive. If sufficient data has been collected, depressing the calibration button will start the computation. After computation, the new parameters will be displayed. These can be rejected or adjusted directly. For advanced users, the parameters can be edited directly and form the starting point for the minimization. The inclusion of the 9'th parameter, "skew" can be toggled using the check-box. Note that for standard windows video stream users, the parameters "fy" will be negative. This is correct and reflects the fact that windows indexes it's images from the bottom left rather than the top left of the image.

Loading and Saving Calibration Parameters

The camera parameters can be saved using the main menu or the load () or save () icons in the tool bar, and subsequently loaded into any computer vision program. For users of the Mixed Reality Software Development Kit (MXRToolkit), easy routines to load in these parameters are provided. For users who wish to use the parameters in their own programs, full details and example code to load in the parameters is provided in the appendix. If the calculated parameters are seem inaccurate or wrong, a new set of "reasonable" parameters can be generated by pressing the new icon ().

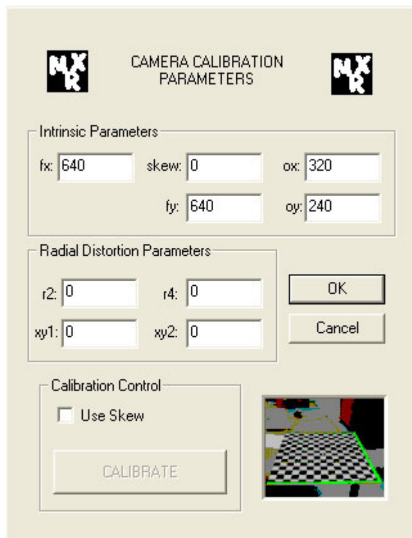


Figure 5. Main calibration dialog box. Each of the current values of the parameters is shown. If enough data has been collected, a calibration can be performed by pressing the calibrate button. If the results are acceptable, the user should press OK, if not then the cancel button will restore the previous values. The results can be directly edited by filling in the boxes. This is useful if the radial distortion parameters are so incorrect that frames cannot be extracted from the video stream.

Creating your Own Calibation Patterns

It is quite possible to create your own chequerboard patterns for use with the calibration software. The following guidelines should be adhered to.

- The border should be either black or bright green
- The border should be u-shaped and cover exactly half of the outer squares
- The number of squares entered into the startup dialog box should include the squares which are half-covered by the border.

Troubleshooting

Frames not being Captured -

There are two major reasons why frames might not be successfully processed by the camera calibration tool. Firstly, it is possible that the position of the chessboard was not correctly identified because contour was not found. To check whether this is the case, select "Threshold" from the Options menu. In order for the contour to be found, it must appear as a continuous white region that is not connected to the background. If this is not the case, the camera settings or the lighting should be adjusted until this is the case.

The second reason why the frame grabbing might fail is because the radial distortion parameters are wrong. If this is the case, then the straight edges of the chessboard will appear as curved. The user can attempt to rectify this problem by directly changing the radial distortion parameters in the in the calibration dialog. A sensible range for these numbers is ranging from -0.15 to 0.15. On return from the calibration box, the video display will be updated to take account of these parameters.

Other Problems -

In case of other problems, please feel free to mail our technical support and we will attempt to resolve the issue: mrlab@yahoogroups.com

APPENDIX A: Camera Model

The camera model can be divided into three components:

- linear camera model (intrinsic matrix) – includes information such as the focal length and field of view of the camera
- non-linear or radial distortion component – a model for distortion induced by non-ideal real world lenses
- the position and orientation of camera in the world (extrinsic matrix)

The calibration program estimates the first two of these components, and which are stored in the "mxrCamera" c-structure, which consists of

```
struct mxrCamera{
	double fx;				// intrinsic – focal length x
	double fy;				// intrinsic – focal length y
	double ox;				// intrinsic – x offset
	double oy;				// intrinsic – y offset
	double skew;		`		// intrinsic – skew parameter
	double r2;				// radial distortion – 2nd order
	double r4;				// radial distortion – 4th order
	double xy1;				// radial distortion – tangential #1
	double xy2;				// radial distortion – tangential #2
	. . .
};
```

The first five parameters describe the *intrinsic matrix* or linear camera model. In an ideal camera of unity focal length, placed at the origin with the optical axis lined up along the positive Z axis, a 3D point in the world (X,Y,Z) projects to the points (x_{cam},y_{cam}) by the equation:

$$X_{cam} = X/Z;$$
$$Y_{cam} = Y/Z;$$

However, real world cameras do not have unity focal length. Moreover, the point (0,0) in pixel co-ordinates is not in the centre of the image, but the top left. The intrinsic matrix converts from ideal camera co-ordinates (x_{cam},y_{cam}) to image co-ordinates (x_{im},y_{im}).

$$x_{im} = f_x * x_{cam} + skew * y_{cam} + o_x$$
$$y_{im} = f_y * y_{cam} + o_y$$

These parameters are given real world interpretations as follows:

- fx is the focal length in the x direction expressed in pixels
- fy is the focal length in the y direction expressed in pixels
- ox is the x offset of the centre of the image from the top-left corner
- oy is the y offset of the image centre from the top-left corner
- skew describes the "skewness" of the CCD array with respect to the camera axis

The parameters, r2, r4, xy1, xy2 describe the non-linear or radial distortion component of the model. In a real-world camera, the linear model generally doesn't apply, but a simple distortion of the camera co-ordinates before applying the intrinsic matrix corrects most of the problems.

$$x_{dist} = (1+r_2*r_{cam}^2+r_4*r_{cam}^4)*x_{cam} + 2*xy_1*x_{cam}*y_{cam}+xy_2*(r_{cam}+2*x_{cam}^2)$$
$$y_{dist} = (1+r_2*r_{cam}^2+r_4*r_{cam}^4)*x_{cam} + xy_1*(r_{cam}^2+2*y_{cam}^2)+2*xy_2*x_{cam}*y_{cam}$$

where:

$$r_{cam}^2 = x_{cam}^2+y_{cam}^2$$

APPENDIX B: C/ C++ Code to Read in Parameter File

Below is a complete listing of all the code needed to read the parameter file into your own program. For users of the Mixed Reality SDK (MXRToolkit), routines are already included to perform this function. The data is loaded into the mxrCamera structure, which contains the parameters themselves and two other structures which can be ignored in this context. Text files containing this code are also included in the main folder.

Listing #1 – Header file – "mxrCameraRead.h"

```
#ifndef MXR_CAMERA_READ_H
#define MXR_CAMERA_READ_H


struct mxrTransform{                                    // 3 x 4 Euclidean transformation matrix
        double r11;
        double r12;
        double r13;
        double tx;
        double r21;
        double r22;
        double r23;
        double ty;
        double r31;
        double r32;
        double r33;
        double tz;
};


struct mxrProjMat{                                     // 3 x 4 Projection Matrix
        double p11;
        double p12;
        double p13;
        double p14;
        double p21;
        double p22;
        double p23;
        double p24;
        double p31;
        double p32;
        double p33;
        double p34;
};


struct mxrCamera{                                      // Main camera structure
        double fx;
        double fy;
        double ox;
        double oy;
        double skew;
        double r2;
        double r4;
        double xy1;
        double xy2;
        mxrTransform T;
        mxrProjMat P;
};

#endif
```

Listing #2 – C/C++ Source File– "mxrCameraRead.cpp"

```
#include "mxrCameraRead.h"

// LOAD CAMERA STRUCTURE FROM DISK
bool    mxrCameraRead(mxrCamera *camera, char *filename, int width, int height){

        FILE *fp;

        // OPEN FILE
        if( (fp  = fopen(filename, "rb" )) == NULL ){
                return(0);
        }

        // READ IN VERSION NUMBER
        int versionNumber;
        if (fread((void *)&versionNumber,1,sizeof(int),fp)<sizeof(int)){
                fclose(fp);
                return(0);
        }

        // CHECK VERSION NUMBER IS ONE
        if (versionNumber!=1){
                fclose(fp);
                return(0);
        }

        // READ IN CAMERA STRUCTURE
        if (fread((void *)camera, 1,sizeof(mxrCamera), fp)<sizeof(mxrCamera)){
                fclose(fp);
                return(0);
        }

        // CLOSE FILE
        fclose (fp);


        // RESCALE PARAMETERS BASED ON CURRENT IMAGE WIDTH AND HEIGHT
        camera->fx*=(double)(width)/640.0;
        camera->ox*=(double)(width)/640.0;
        camera->fy*=(double)(height)/480.0;
        camera->oy*=(double)(height)/480.0;
        camera->skew*=(double)(height)/480.0;

        return(1);
}
```

Notice that the camera parameters depend on the current width and height of the video stream produced by the camera. Hence, these parameters must be passed into the routine. The file stores the parameters as appropriate for a 640x480 video stream and rescales them appropriately depending on the current size.