Manual of "Vision"

1 Introduction

Vision is a data acquisition and treatment program for cold atom physics experiments, especially Bose-Einstein condensation (BEC) experiments. It also acts as a sort of automatic lab book keeping protocols of every experimental run executed. It is used on at least five BEC experiments worldwide and was originally developed for the Lithium experiment at the E.N.S. in Paris. The general setup of the computer system used with a BEC experiment using Vision is the following. A control computer, running for example the program "Control" (see

`http://george.ph.utexas.edu/~control/index.html`

) generates the analog and digital waveforms used to steer the experiment. This control computer is linked via ethernet with the data acquisition computer running "Vision". The control computer sends commands to prepare the camera to take images to "Vision" via TCP/IP or the serial port. Vision communicates with the camera and acquires the images. After an experimental run, "Control" transmits the parameters of the experimental sequence to "Vision" which stores them together with the acquired images. Vision treats the raw images and obtains absorption or fluorescence pictures from which it obtains measurements like the atom number or the size of the atomic cloud. You can use any other control system with Vision as long as you can implement the simple communications protocol between Vision and your control program over TCP/IP or the serial port. When you adapt Vision to your system you have to implement support for your camera system. Vision comes built in with support for some Princeton, Andor and Apogee cameras. Some cameras are delivered with drivers which do not work with Borland C++ 5.02 in which Vision is written. For those cameras, eg. the Apogee Alta series, a small helper program supports Vision. It is called "ApogeeServer" and can be downloaded on the webpage given above. It is written in Visual C++ and can thus be adapted to all cameras delivered with drivers compatible with this language. Vision and ApogeeServer communicate over ethernet using TCP/IP to exchange camera commands and picture data.

This document describes the user interface of Vision and its most important features. Then it gives guidelines how to adapt it to your system and implement your own camera driver. Finally the communication protocol between the control computer and Vision is described. With this information it should be easy to adapt Vision to your system.

2 User interface

Visions user interface is divided in several sections. The most important one is the big sub window in which normally the absorption picture of the last data point under consideration is prominently displayed. To the left and below you see profiles of the picture displayed. In these profiles you can see blue and red bars which you can position by using the left and right buttons of the mouse. The rectangular area marked in blue marks the position of the atoms. The area marked in red, but excluding the blue area gives Vision a region where no atoms are present from which it can determine an eventual background offset. The profiles shown are usually the sum of the lines or columns in between the blue lines. Gauss fits are automatically performed to the profiles in the region marked by the red lines. The atom number is determined by integrating the signal in the blue marked area and subtracting an eventual offset obtained from the red region.

The result of the atom number calculation and the Gauss fits are displayed in the window "Gauss and integral fit result". "Dens" is the peak optical density, "X0" and "Y0" give the position of the

center of the gauss fits in millimeters. "sx" and "sy" are the sigma widths of the Gaussians. "Ox" and "Oy" is the value of the offset of the Gaussians. "NG" is the atom number extracted from the Gaussian fit and "NI" the one obtained from integration over the image in the blue marked region. "n0" is the peak density in cm^{-3}. "Fl" the fluorescence count of the MOT just prior to starting the experiment. "NOf" is the offset in optical density extracted from the red region excluding the blue region. "ReFl" is user specific and can be for example the fluorescence after recapture in a MOT from an optical or magnetic trap. "TF/TC" is the ratio between the Fermi and the critical temperature of the gas. "Rx" is the Thomas Fermi radius.

To obtain the latter values, "Control" did transmit the trap parameters used to Vision. They are displayed in the box "Measurement parameters". "Camera" is the number of the camera used for this image. "Det" is the detuning of the absorption beam. "TOF" is the expansion time (time of flight). "Isotop" is the Isotop used for example Lithium 6 or Lithium 7. This was used in the original Lithium experiment for which Vision was created. It can mean completely different things in your experiment for example 6 might mean Potassium and 7 Rubidium or 6 might mean vertical imaging system and 7 horizontal imaging system. Trap is a number describing the type of trap. For trap type 0 = Ioffe Pritchard trap, the next three parameters "B", "G" and "C" give the bias in Gauss, the gradient in Gauss/cm and the curvature in $Gauss/cm^2$. "P1" to "P4" are user defined parameters. If you are taking many experimental points in a series of measurements, those are the up to four parameters which can be varied from point to point. Usually only one parameter is varied, eg. the expansion time and then displayed under "P1". The last line displays a text string describing the experimental run. Each stage of the experimental sequence is given a specific letter and all stages used to obtain a certain data point are chained next to each other giving the shortcut string displayed here. At the end of the string the final trap type is written out in normal words. This descriptive string is generated by "Control" and send over to Vision.

Using the parameters from these two boxes Vision calculates more advanced things and displays them in the box "Temperature and Atom number fit results". Which parameters here are valid for a given data point and how they are calculated depends on the context in which the picture lays and on fits performed on a series of experimental runs. This will be explained later.

In the main absorption picture window you also see a cross cursor. It usually marks the position of maximum optical density in the blue region, unless it has been fixed to stay at a certain position using the menu option "Options→Fix cursor". If you move the cursor around with the mouse, the profiles will be updated and display not any more the sum over the image but the local one pixel line cut at the position of the cross. The position of the cursor is displayed in the box "Cursor Position" under "X" and "Y". "H" is the optical density (height) of the image at the center of the cross. "Xp" is the position of the left blue bar and "Dx" the distance between the horizontal blue bars in millimeters. "Yp" and "Dy" are defined accordingly for the vertical blue bars.

To the right of the main picture the color scale is displayed. You can change it by clicking and moving its upper or lower border with the left button. The right mouse button will display white iso optical density lines.

To the right you find six small windows. The window "Fluorescence" is used for test images taken by using the menu option "Camera→... picture" or "Camera→... film". For absorption images "Raw absorption" displays the probe beam profile with atoms present, "Probe" displays the probe beam profile without atoms, and "Noise" displays the noise present without probe beam from scattered light. The "Absorption" picture is ("Raw absorption"-"Noise")/("Probe"-"Noise"). The optical density picture is -ln("Absorption")/σ_{abs} and thus after the absorption law equal to the 1D integral over the atomic density along the absorption beam. That is why the 2D integral over the "optical density" picture is equal to the 3D integral over the atomic density distribution and thus the atom number. You have the responsibility to enter the correct Clebsch-Gordan coefficient of your atomic transition in the file "setup.h" of Vision so that Vision can calculate σ_{abs}. Clicking on any of the images loads

it to the main picture window and starts the extraction of the parameters from it.

For fluorescence images "Raw absorption" is the fluorescence image with atoms and "Probe" is the background image without atoms. "Absorption" and "Optical density" are both proportional to "Raw absorption"-"Probe". "Optical density" is re scaled so that the minimum and maximum signal just span the whole color scale, whereas "Absorption" is scaled using a value for the maximum absorption transmitted to Vision by "Control".

The twelve images to the right display the main image of the last six experimental runs. You can use the left and right column as you want, eg. for vertical and horizontal cameras or for different isotopes. The picture actually displayed here and in the main window is selected with the menu option "Show→...". The last acquired or loaded image is marked by "¿". The filename of the data point is marked below the picture. Clicking on any of those pictures loads the corresponding data point to the main picture.

Below the six raw images a white region displays the result of the measurement done over the parameter varied during the measurement. The parameter(s) displayed is selected with the checkboxes to the left of the frame. The checkboxes on top select if only data belonging to vertical or horizontal images or both is displayed. These checkboxes also determine which image should be the one displayed after acquisition in the main window. Below the graph the name of the parameter varied is displayed. That's usually the picture number. But in a measurement series it is one of the parameters varied, eg. the expansion time. A data point is loaded when you click on it and you can load the next or the previous data point with the two arrow buttons at the lower right side of the box. You can mark a point as invalid by clicking on it with the right mouse button. A point is deleted from a series by loading the point by clicking on it and the using the menu option "Measurement→Delete measurements". A single data point is loaded with "Measurements→Load measurement".

The menu "Series" gives you more options to do with measurement series. "Load" loads a series from hard disk, CD or DVD. "Add" adds another series to the ones loaded already. "Save" saves the points loaded. "Shift parameters" toggles to the next parameter varied for series during which multiple parameters have been changed. "Sort series" sorts the series after the value of the varied parameter. This is useful since during measurements usually the order of the parameter values measured is randomized to fight experimental drifts. "New series", "Start series" and "Stop series" enables you to create series manually, which means that data points are stored in their proper subdirectory on hard disk.

Simple fits can be performed to the data displayed using the "Fit" menu. But it is recommended to load the ASCII files describing the experimental series with a program like "Origin" and performing the fits there.

Below the data point window the lab book window is displayed. It contains a list with a description of all data points measured that day containing the name of the data point and the names and values changed since the last data point. When loading a new data point this list automatically scrolls to the description of this point if possible. Here an example for the description of a datapoint:

```
*P-0011A7 MOMP Magnetic trap
 ExpansionTime=8.000 (5.000)
```

"P-0011A7" is the filename and tells you that it is the eleventh point of its series taken with the horizontal imaging system (or Lithium 7, or whatever you defined 7 to signify). The atoms come from a MOT "M", underwent optical pumping "O", where captured in a magnetic trap "M" and an absorption picture was taken of them "P". The last stage before picture taking was a Magnetic trap. The only parameter varied since the last image was the "ExpansionTime" and it went from 5 ms to 8 ms.

Below this window you have buttons to steer 2D fits and 2D fit results. This subject is a bit involved and not often used anyway so I won't bother to describe it further unless somebody asks me

for it.

At the very bottom of the window you can see a status bar, displaying the messages exchanged between Control and Vision or fit results.

We have discussed already most of the menu options. Here are the remaining ones. In the "File" menu you will find export options for the image displayed in the main window. "Save as matrix" saves the image as an ASCII table, the others as BMP or TGA images. "Save low byte as BMP" saves only the lowest 8bit of the image and can be used to measure the camera noise. "Save Profile" saves ASCII tables of the profiles.

"Calc" contains commands to execute on all pictures on the series which is loaded at the moment. The only important ones are "ReCalc" which recalculates the fits of each image and "Create film" which creates a subfolder into which BMP files of each main window frame are exported with a filename that makes importing to a program like "GIF construction set professional" easy to create GIF animations.

There are various fits in the "Fit" menu, of which the most important one is "Fit temperature" which obtains the temperature of a cloud from its expansion behavior.

The most important options in the "Options" menu are the "Fix cursor" and "Zoom" commands. Sometimes it can be useful to alter the standard behavior of the gaussian fits to the profiles by allowing a slope or disabling the fit offset which can be done under "Options→1D Fit options".

3 File system

The main directory for data storage is declared in "setup.h". This can be "C:\Rubidium\Data". Here you find "Logbook.dat" describing all experimental series ever executed with your system. Also reference images for the probe beam profile and the noise images are stored here, in case you chose the option to take only the image with atoms to speed up data acquisition. The file "Message.txt" contains the last message send to Vision by Control and can be looked up over the LAN to see where Vision is. You can also connect a remote version of Vision over LAN to the master Vision and you get a duplicate of the Vision screen contents on a remote computer in your LAN. Vision creates a new subfolder for each day, eg. "C:\Rubidium\Data\04May07". In this subfolder "DataP-7.dat" and "DataP-6.dat" contain ASCII tables describing the measurements you did. Load these files with a program like "Origin" to fit or print your data. "Logbook.txt" contains the logbook displayed in the logbook window of Vision. Each data point consists of several files. "P-0001_6.txt" and "P-0001_7.txt" contain all the parameters used for the data point (here data point number 1). Those files are very useful. For example if you want to reproduce a data point but something doesn't work and you don't know which one of the hundreds of parameters describing your experimental sequence has changed, use a program like "CSDiff" to compare "P-xxxx_6.txt" with "P-yyyy_6.txt" and you will find the difference instantly. Next a series of TGA files like "P-0001A6.TGA" contain the raw 16bit pictures. "P-0001A7.bmp" and "P-0001A6.bmp" contain 8bit bitmaps of the main window picture, normally the optical density picture. Put Windows Explorer in thumbnail display mode and you have a nice overview of all pictures you did a certain day. For each measurement series a subdirectory like "S1" or "S2" and so on is created and the same types of files are stored here. For the creation of films directories "F1" and so on are created.

4 Setup of Vision

For the standard installation of Vision, create the directory "c:\Rubidium\" and extract "Vision2.zip" into this directory. A subdirectory "C:\Rubidium\Vision" is created, containing a source subdirectory,

the manual and so on. Put windows in a 32bit color screen mode with at least 1152x864 pixel resolution. Create the file "C:\Rubidium\Vision.cfg" containing:

```
DataDisk=2
DataDirectory=c:\Rubidium\Data
MainDirectory=c:\Rubidium
HardwareAccess=1
LiveDirectory=F
LiveDisk=5
LiveConnectTime=10000
```

If you do not want Vision to access the hardware, set HardwareAccess to 0. This is good for a first test. DataDisk is the hard disk used for the data "2=C:\ 3=D:\ and so forth. DataDirectory and MainDirectory are the names of the directories used by Vision. LiveDirectory is the locale network disk drive name (here F signifies F:\) over which a remote Version of Vision accesses the lab computer. LiveDisk is the same but in the numbering scheme 1=a: and so forth. LiveConnectTime is the time in seconds that the remote computer stays connected with the lab version of Vision after selecting the "Options→Live Connect" menu option. (To make a live connection, create a network drive with windows explorer (in this example drive F) and link it to the data disk drive of your lab data acquisition computer. Install vision on the remote computer with HardwareAccess=0. Run Vision, select "Options→LiveConnection".)

Next you need to modify the parameters in "setup.h" for your system. This includes selecting the mass and Clebsch-Gordan coefficient of your atom, selecting your camera system, configuring the directories in which you want to store your acquired data and eventually the TCP/IP address of the computer running "ApogeeServer". Here is a typical version of setup.h. After modifying "setup.h" you need to recompile at least "visiondg.cpp" for the modifications to take effect and for some modifications you even need to rebuilt all.

```
#if !defined(__setup_h)
#define __setup_h

//The wavelength of the probe laser
const double Lambda=780E-9;
//gF used
const double gF7Li=0.5;
const double gF6Li=0.5;
//mass in mp
const double MassLi7=87;
const double MassLi6=87;
//maxium optical density used in display
const double MaxOptDensity6Li=3;
const double MaxOptDensity7Li=3;
//Clebsch Gordan coefficients
const double ClebschLi7=1.0/2.0;
const double ClebschLi6=1.0/2.0;
//Linewidth (not really important)
const double GammaLi7=122E6;
const double GammaLi6=122E6;
const double EichHeight=1/0.7;
const double EichOffset=0.1;
//ellastic collision cross section
```

```
const double SigmaEl=6.9E-16;
const double CameraInclination=0;
const double OffsetButtonScale=0.01;
const double InitialOffsetShift=1;

//select the camera you are using
//#define AndorCameraUsed
//#define MatroxCameraUsed
//#define ApogeeCameraUsed
#define NetAltaCameraUsed
#define NetPrincetonCameraUsed

//parameters required by the cameras, eg.
//init file names, IP addresses of camera
//computers, IP ports
const bool MatroxCameraExtern=true;
unsigned short MatroxCameraPort=2222;
char* MatroxCameraIPAdresse="129.199.118.201";
char* MatroxCameraComputerName="Enrico";

unsigned short NetPrincetonCameraPort=2654;
char* NetPrincetonCameraIPAdresse="128.83.131.195";

unsigned short NetAltaCameraPort=703;
char* NetAltaCameraIPAdresse="128.83.155.53";

const char* ApogeeInitFile="c:\\Rubidium\\ap47p.ini";

//Selects if the control computer is speaking to Vision
//over the serial port or over ethernet
const bool ControlComputerSerial=false;
//If ethernet is used, this is the IP port of Vision
unsigned short ControlComputerPort=701;

//The config file name and initial directories
const char* ConfigFileName="c:\\Rubidium\\Vision.cfg";
char* InitMainDirectory="c:\\Rubidium\\";
const char* InitDataDirectory="c:\\Rubidium\\Data";

const bool InitHardwareAccess=true;
const char InitLiveDirectory='E';
const int InitLiveDisk=4;
const int InitDataDisk=2;
const double InitLiveConnectTime=10000;

#define Li7Name "Horizontal"
#define Li6Name "Vertical"
const filmtime=500;
const testserialtime=500;

//Parameters used for the layout of the main window
```

```
//not really necessary to modify
const twopicturelists=1;
const WindowSize=1152;
const WindowHeight=839;
const R=10;
const PH=150;
const PH2=150;
const PH3=150;
const D=5;
const D2=10;
const WH=256;
const WV=256;
const BB=16;
const Slash=6;
const Text=10;
const SB=25;
const SH=25;
const PictureListHeight=740;
const BoxWide=60;
const DX0=R+PH+D+SB+D2+WH+D2+BoxWide+D2;
const TH=16;
const D3=5;
const PB=(WV-2*(TH+D3))/2;
const TBW=PH+D+SB+D2+WH;
const Y1=R+WV+D+SH+D2;
const Y2=R+WV+D+SH+D2+PH+D2;
const MTextBoxH=91;
const GTextBoxH=91;
const TTextBoxH=91;
const X1=R+PH+D+SB+D2+WH+D2;
const TextBoxHeight=18;
const TextBox3Width=150;
const TextBox4Width=115;
const TextBox4WidthSmall=95;
const InitialOptDensMethod=0;
const PictureListColumns=1;

#endif
```

After having adapted "setup.h" for your system, you select "Project→Build all" in Borland C++ and run Vision. Now copy the test data set which you can copy from our webpage to "C:
Rubidium
" and load the test data with "Series→Load series". Now you can explore the options of Vision.

5 Adding a new camera driver

Essentially you look which camera driver that already exists is nearest to the one you want to use and then you modify the files corresponding to this driver. Say you have a slow scan camera for which you have Borland C++ compatible drivers, then you might want to just change the file "Andor.cpp" and let your camera appear to the rest of Vision like the Andor camera. If you do not have Borland

C++ compatible drivers, but Visual C++ compatible drivers, you need to use "ApogeeServer" and modify it for your camera using the manual provided with "ApogeeServer". To the rest of Vision your camera appears as the NetAlta camera.

6 Communication with Control

It is easiest just to use "Control" to control your setup. If you do not want to do that, then download "Control" all the same from my webpage and look at the CVision class. It contains all commands necessary to communicate with vision. Look at the CSequence class and search for all calls of CVision to understand how Control talks to Vision. Essentially before each image is taken a command "VisionSetCameraParameters" is send setting up the camera and then a "VisionTakeAbsorptionPicture" command starts the image acquisition sequence. One parameter is a bit special, "TriggerDelay". It determines the amount of time Vision waits after having received the "VisionTakeAbsorptionPicture" command before sending the software trigger command to the camera driver. This keeps the camera as long as possible in the CCD flushing mode. The parameter "APNoiseMode" determines which of the three images used for an absorption image are really taken. 2 means all 3 images including noise image, 3 means only absorption with atoms and probe profile, 4 means only absorption with atoms. Vision loads the non taken images from hard disk. It takes the last ones of the sort missing recorded. Look up TVisionDialog::AbsorptionPicture() to understand exactly what is happening during the image acquisition. "VisionStartSerie" and "VisionStopSerie" commands can start and stop a new measurement series which is stored in a new sub directory. VisionStartSerie expects the number and names of parameters varied and a comment string as parameters (look it up in TVisionDialog::CheckSerial() in "visindg.cpp").

You can get a list of possible commands by looking through TVisionDialog::CheckNoArgCommands() and TVisionDialog::CheckSerial() in "visindg.cpp". Here you can also declare your own additional commands.

Here is a protocol of communication between "Control" and "Vision" to take one image. Messages marked with $\gg$ are send from "Control" to "Vision". $\ll$ marks the other way round. All messages start with "*" and end with "#" even when not marked as such in this example. "//" marks comments which do not actually appear during the communication.

```
// check if Vision is there
>> *VisionReady#
<< Ready
// check if camera is ready
>> *VisionCameraReady#
// Vision tells us it is ready to receive parameters:
<< Ready
// Number of camera:
>> *       8#
// ok, camera is ready:
<< VisionCameraReady
// Main picture display mode
>> *VisionMainPictureOpticalDensity#
<< Ready
// Set the camera parameters:
// Look up Visiondg.cpp and search for VisionSetCameraParameters
// to understand their meaning, or look up CVision.cpp in
// "Control"
```

```
>> *VisionSetCameraParameters#
<< Ready
>> *       8#
>> * 0.00300#
>> * 0.00300#
>> *       0#
>> *       0#
>> *    1024#
>> *    1024#
>> *       1#
>> *       1#
>> *20.00000#
>> *-30.00000#
// Now take the picture:
>> *VisionTakeAbsorptionPicture#
<< Ready
>> *       0#
>> *       8#
>> *P#
>> *-#
>> *       6#
>> *       0#
>> *       0#
>> *16384.00000#
>> * 2.00000#
>> * 2.00000#
>> * 0.00000#
>> * 0.00000#
>> * 5.00000#
>> * 5.00000#
>> * 1.00000#
>> *1364.57600#
>> * 0.00000#
>> *       0#
>> *       2#
>> *MoOMCP Magnetic trap#
<< Ready
<< Ready
// Picture has been taken, now send parameters
>> *VisionSynchronizeParameters#
<< Ready
// since this is first image, send all
// parameters, later send only changed ones
// Vision Asks for all by sending *SendAll# and
// for the changed by sending *SendChanged# at
// this moment.
<< SendAll
// These parameters are send in inverse order in which
// they will be saved in the "P-0001A7.txt" files.
// a double parameter marked with D
>> *DVerticalSheetPositionMax=45.000#
```

```
>> *DVerticalSheetPositionConversion=0.763#
>> *DVerticalSheetCenterFrequency=100.000#
>> *DHorizontalSheetPositionMax=50.000#
// hundreds of more parameters
.
.
.
>> *DMOTEarthFieldCompensationCurrentZ=0.315#
>> *DEarthFieldCompensationCurrentY=1.500#
>> *DEarthFieldCompensationCurrentX=0.820#
>> *DMOTAOMIntensity=0.000#
// a string parameter marked with S
>> *SDebugSingleAtomDetectionFilename=c:\Rubidium\SingleAtom.dat#
// an integer parameter marked with I
>> *ITEM01LatticeNrSheets=3#
// a bool parameter marked with B
>> *BUseVacuumChamberShutter=FALSE#
>> *BStartWithTinyMOT=FALSE#
// a title of a new parameter menu marked with T
>> *T*** MOT,CMOT, Molasses parameters ***=#
// These where all the parameters
>> *VisionSynchronizeParametersEnd#
<< Ready
// Now send some data measured by control
// during this experimental run like MOT
// Fluorescence. Look up "Control"
// CVision::SendPictureData
// to understand meaning.
>> * 0.20000#
>> * 0.00000#
>> * 0.00000#
>> *28.00000#
>> * 0.00000#
>> * 0.00000#
>> * 0.00000#
>> *186.48000#
>> * 0.00000#
>> *2000.00000#
>> * 0.00000#
>> * 0.00000#
>> * 0.00000#
>> *      10#
>> * 0.00000#
>> *72790.76500#
>> * 0.00000#
>> * 0.00000#
>> * 1.00000#
>> * 0.00000#
>> * 0.00000#
>> * 0.00000#
>> * 0.00000#
```

```
>> * 0.00000#
```