KISMET V 6.0.2

User Manual

Version 1.4

Dr. Uwe Kühnapfel
Hans-Georg Krumm
Markus Hübner
Christian Kuhn

Forschungszentrum Karlsruhe GmbH

Revision: July 23rd, 1998

I Abstract

KISMET (**Ki**nematic **S**imulation, **M**onitoring and Off-Line Programming **E**nvironment for **T**elerobotics) is a 3D graphical simulation program, which has been under developement at Forschungszentrum Karlsruhe (FZK) since 1986. KISMET is suitable for a wide range of applications including

- remote handling in dangerous environments,
- robtics simulation, programming and monitoring,
- scientific visualisation
- medical date visualisation and VR-based surgery training

This **KISMET V6.0.2 User Manual** is designed to give you the basic information about using the KISMET simulation environment. It is mainly intended to make you familiar with the man-machine interface (MMI) and the basic concepts of its use.

Following topics are covered in particular:

- An introduction to KISMET and its general features
- The basic display layout
- KISMET module structure, program startup and commandline options
- UNIX environment variables
- Fundamentals of the user interface
- Menu commands (the menu-tree) using the Menu-Panel MMI and - in full-screen mode only - the Popup-Menu MMI
- Function-Key commands
- How to use the 2D-editor 2Ded

II Preface

Note: Your installation and/or the data-carrier (cardridge tape, DAT, CDROM etc.) attached to this notice contains a copy of the program KISMET and some models for demonstration purpose only.
The ownership of this software and all attached documentation still belongs to **Forschungszentrum Karlsruhe GmbH** (FZK), unless your organisation has a license agreement with FZK. Software and documentation may not be resold, published, copied, duplicated or disclosed to third parties, in whole or in part, without the prior written consent of FZK.

III Table of Contents

IV List of Figures

V List of Tables

1 Scope of this document

This KISMET User Manual is designed to give you the basic information about using the KISMET simulation environment. It does not attempt to teach you how to write C-programs, or to bring you to an expert level in robot programming, CAD or mechanical engineering. Rather, it is intended to make you familiar with the KISMET man-machine interface (MMI) and the basic concepts of its use.

KISMET users in the expert class, or those involved in robotics control system integration, may find that this document lacks the depth of information they need. For them we recommend to read the references [1],[2] and [3].

To use KISMET, knowledge is assumed, as follows:

- use of terminal and graphics input devices
- use of a UNIX system editor
- operation of the Silicon Graphics window manager
- UNIX system directory/file structure
- principles of robotics, computer graphics, kinematics, and robot-programming

The material in this guide is organized, as follows:

Chapter 2 What is KISMET ?	Gives a brief overview of KISMET, its purpose, background and general features.
Chapter 3 KISMET Display	Explains the KISMET screen-layout.
Chapter 4 Program and User Environment	Describes the KISMET Module structure and gives information about the different filetypes, the directory structure and important UNIX environment variables.
Chapter 5 The KISMET User Interface	Includes information about the fundamentals of the KISMET man-machine interface (MMI) and describes the command-line options when you start KISMET.
Chapter 6 List of Commands-Chapter 17 OPTIONS	Describes in detail all menues and buttons visible for the user while executing KISMET in the non-full-screen mode. This chapter can be used by novice users to get an overview of KISMET operation, for experienced users it gives a reference to all available menu commands.
Chapter 19 Fast Keys	Informs about the Fast-KEYS on the keyboard, and how they are used in KISMET.
Chapter 21 The 2Ded-Editor for SWEEP Operations	Gives information on the 2D model editor 2DED, which is used for some primitives (sweeps) during 3D-GEOMETRY-Editing in KISMET. 2DED is iplemented as external module of KISMET, it can also be used as a standalone program.

Chapter 22 References and Literature	Includes a list of reports, books and papers, which are either reffered to in this document, or which we recommend for further reading (for those users, which like to become real experts).
Appendix•A - List of implemented IRDATA-Commands	Includes a list of executable IRDATA commands in KISMET

2 What is KISMET ?

2.1. General features

KISMET (**Ki**nematic **S**imulation, **M**onitoring and Off-Line Programming **E**nvironment for **T**elerobotics) is a software tool for effective planning, simulation, programming and monitoring of remote handling equipment, industrial robots and various types of mechanisms.

The program system KISMET was initially developed at the *Forschungszentrum Karlsruhe GmbH (FZK)* formerly known as *Kernforschungszentrum Karlsruhe (KfK)* for use within the EC programme for development of controlled thermonuclear fusion (JET, NET), and other nuclear oriented remote handling applications in hazardous environments.

Thus, the development work aimed at supporting operators and planners of handling equipment during preparation and execution of remotely controlled tasks in plant areas that are inaccessible for humans or difficult to observe. Execution of these teleoperation tasks is supported through **synthetic viewing**. The operator orients himself within the working environment by means of camera observation (if available) as well as animated 3D computer graphics provided by KISMET. This combination of computer graphics and video technique is also called integrated viewing. For this type of applications, the graphical scene presentation has to be as realistic as possible and at the same time as fast as possible.

KISMET is implemented on a high-performance graphics workstation and allows via window techniques for several scene views at the same time with interactively selectable levels of detail. Apart from these application areas typical within nuclear research, KISMET is, however, also suitable for planning and programming in industrial automation and manufacturing applications. Thus, it allows fast modelling, analysis, and modification of simulated manufacturing cells.

Since KISMET is oriented towards neutral file formats for robot programming (IRDATA, ICR in preparation) and for geometry data transfer (*Inventor* format, STEP processor available), it is more easily being integrated into existing CIM concepts.

KISMET allows for a real-time, synthetic generation of any view of handling and manufacturing cells in shaded and textured display (Gouraud shading with surface textures), as wireframe, or as transparent models. The rendering mode can be set interactively as a display attribute for any single geometry entity and/or for assembly parts. The integrated raytracing module can produce photorealistic images (non-realtime).

KISMET was written in the "C" language and is presently available for Silicon Graphics workstations running IRIX 5.3. or higher. A Windows-NT version is ready as a Beta-release version (July 1998). The KISMET software is implemented using OpenGL graphics library.

In the implemented version V6.0.2, as of Juli 1998, mechanical structures are simulated with rigid links, and optionally, robot dynamics can be modelled and simulated in realtime, using the "Recursive NEWTON-EULER-Dynamics reffered to Link Coordinates" method (sometimes referred to as LUH, WALKER, PAUL algorithm). Additionally, the dynamics module allows to model the control characteristics of the robot system. Another functional option is modeling and realtime-simulation of elastomechanic effects, like robot link torsion and bending. This feature is using reduced stiffness matrices, which can be obtained from a FEM-program.

Using KISMET, it is possible to model and simulate in realtime **tissue-elastodynamics** using a spring-mass system modelling approach (full elastodynamics). For execution of large dynamic systems, additionally a **Fast Finite-Element Model** (FFEM) approach was implemented in

1998. The latter simulation modes are used mainly for medical applications, for example surgery simulation and training. Another field of medical applications is the visualisation of real-time visualisation of Voxel-based volumetric datasets, such as CT-, MRI-scans.

The concurrent simulation of any number and types of robots and other mechanisms, including tree-like mechanical structures and planar closed-loops, is possible in KISMET.

2.2. KISMET Operating Modes

The "**Off-Line**" operating mode serves for:

- creation of workcell layouts,
- editing of geometry, kinematics and workcell topology
- operator training
- remote handling operation planning
- accessibility tests
- creation (textually, graphically), simulation, verification and execution time measurement of robot programs
- numerical test for collisions
- creation of animation sequences

While in "**On-Line**" (supervisory mode, monitoring mode) the position sensors (resolver data) of the robot control units or other mechanical systems, such as VR- and surgical input devices, are used to drive the kinematic simulation model in KISMET and to present a realistic and problem suited display of the task area. Therefore the graphical model configuration (robot joint position data, mounted tools, position or state of installed objects in the working environment) is controlled through the real running process. Hereby, the operator can specify any kind of view in the handling environment or manufacturing cell. While in this operation mode, the operator can supervise automatically running robot programs, or directly control the equipment by means of manual operation controls (man in the loop control).

KISMET supports the operator in this mode in terms of spatial scene presentation and, additionally, automatic collision warnings.

2.3. KISMET Data Structures

In KISMET a hierarchical, recursive, tree like data model is used. The user can specifiy at any time and interactively, the level of detail to be displayed. Components or model parts can be activated or deactivated for display at any time. This allows to display higher levels of detailing without loss of realtime performance.

Robots and other handling equipment or mechanisms (including tree-like mechanical structures and kinematically determined planar closed loops as typically found in hydraulic drives, gearings, crank mechanisms etc.) can be defined interactively, or can be imported from CAD systems (a STEP processors is available). Dedicated data-import interfaces exist for *SoftImage* models and for the *Inventor 2.0* (also VRML) exchange format.

KISMET provides a high modelling functionality for **geometry**:

- 3D volume geometry primitives like BOX, SPHERE, CYLINDER, CONE, ROTATIONAL_SWEEP, LINEAR_SWEEP, POLYHEDRON and PIPE (primitive for piping). KISMET generates for efficient display an internal surface data representation (BREP-model) during model startup.
- Additional line primitives CURVE (parametric polynomial definition, VDAFS, DIN

66301), LINE, and POLYGON.
- Polynomial surface primitive SURF (VDAFS, DIN 66301) and non-uniform rational B-spline surfaces (NURBS).
- Access to the "exact" geometry definition is always possible.
- Parametric objects can be created to simulate for example an extending crane cable, laser beams, springs, rubber parts etc.
- Boolean modelling operations (UNION, DIFFERENCE, INTERSECTION). The resulting shape from these operations is stored as a POLYHEDRON model
- 3D-elastodynamical objects (NURBS and polygonal surface geometry), using a volumetric modelling approach.
- Voxel-based volumetric datasets

Features for **kinematic** modelling are:

- Revolute and prismatic degrees of freedoms are used as basic elements for kinematic modelling. Higher degree mechanisms like spherical, cylindrical, etc., joints are built as combinations of these basic elements.
- Kinematical chains may have any lenght (number of successive DOFs) and may have any number of branches (chain offsprings).
- Kinematically determined planar closed-loop mechanisms can be defined.
- For a set of kinematical structures, the inverse kinematic problem is solved using an internal library of subroutines. This library provides the analytical solution for a specific class of robots.
- Additionally a "Generalized Inverse Solution" is implemented in KISMET, using the "Modified NEWTON-RAPHSON" (MNR) method. The Jacobian is calculated numerically in the implemented algorithm, i.e. there is no additional programming necessary from theuser. The implemented algorithm provides not only a solution for robots with 6 joints or degrees-of-freedom (DOF), but also for the kinematically underdetermined case (less than 6-DOF) and the overdetermined case (more than 6-DOF, robots with kinematic redundancy)

Features for **multibody-dynamics** and **robot control** modelling are:

- realtime multibody-dynamics simulation with direct and inverse dynamics solution(s).
- Elastostatic deformation of large bodies due to gravity and load using a beam-bending model.
- Robot controller simulation providing modelling elements for joint-level control (P-, PI- and PID-control blocks).

General features:

- Since all definition files are stored in ASCII-format, neutral file format processor are easily developed.
- Every user may define his/her own component and robot libraries.
- Material- and colour tables are user definable.
- There is no logical limit in model size or detailing level.
- TV-CAMERA simulation and control algorithms (automatic tracking of any objects or end-effectors, fast positioning through inverse camera-actuator kinematics).

2.4. KISMET-MMI

KISMET is operated mainly via the mouse and keyboard driven KISMET User Interface. Additionally, other input devices are integrated into the KISMET-MMI concept:

- Silicon Graphics "Dials-and-Buttons-Box"

- 6-DOF Robot-Teach Ball
- Space Mouse

Display features:

- Multiple viewports with perspective or orthonormal projection
- Variable realtime object rendering: flat-shaded, goraud-shaded, hidden-line, wireframe and transparent rendering. The drawing attribute can be set individually for any geometrical object or parts assembly in the szenario
- Photorealistic image synthesis using the Raytracing method (non-realtime)

2.5. Simulation and Off-Line-Programming

For simulation of robot programs an IRDATA interpreter was integrated in KISMET (DIN 66313, part 1). Also available is a PASRO to IRDATA compiler (PASIR) for textual robot off-line programming. KISMET provides the following functionality for off-line programming and simulation:

- Parallel and concurrent simulation of multiple robot confurations is possible. For synchronisation of different system components there are statements available for 'wait', conditional program execution and program flow control (while, for, if-then-else etc.).
- The analysis of PTP (synchronous and asynchronous), linear and circular motions can be done quickly. For the calculation of time coordinated motion trajectories, acceleration and velocity profiles of single axes, robot arm configuration, motion restrictions are included in the simulation model.
- Robot program simulation can be executed
    - in "realtime", i.e. the simulation in KISMET takes the same time as the execution of the "real" robot program
    - with an operator specifiable sampling interval
- In the latter mode, the calculation of robot program execution time is possible.

Additionally to the direct, menu driven user MMI, KISMET offers a textual **KISMET-SCRIPT** command language. SCRIPT commands can be entered into KISMET in different ways. If read in from an ASCII file, very complex animation sequences can be created.

However, SCRIPT commands can also be entered into KISMET be means of UNIX interprocess communication techniques. KISMET provides a 'message-queue' channel for other, concurrently running processes as a command interface (faster than 'pipes'). Using this method, each user can build his own MMI around KISMET.

SCRIPT-commands can also be entered as comments into IRDATA-programs. When these comments are sent back from the real robot controller during on-line program execution, KISMET can reconfigure its data structure upon process events or configuration changes (e.g. tool or end-effector exchange, grabbing of handled objects etc.).

2.6. KISMET Graphics Performance

The allowable performance minimum refresh rate (different images per second) depends on the application and model size but is normally more than 8 to 15 per second (Gouraud shaded, Z-buffered, textured, one scenario viewport, 25000 polyfaces, 50-200 independent modelling frames on a *Silicon Graphics OCTANE/MXE, R10000, 250 MHz* workstation). Implemented applications have demonstrated that this target value is achievable with KISMET - with some model optimisation efforts - even for much larger model sizes.

3 KISMET Display

The KISMET standard screen is shown in *Figure 1: KISMET Display Layout.*

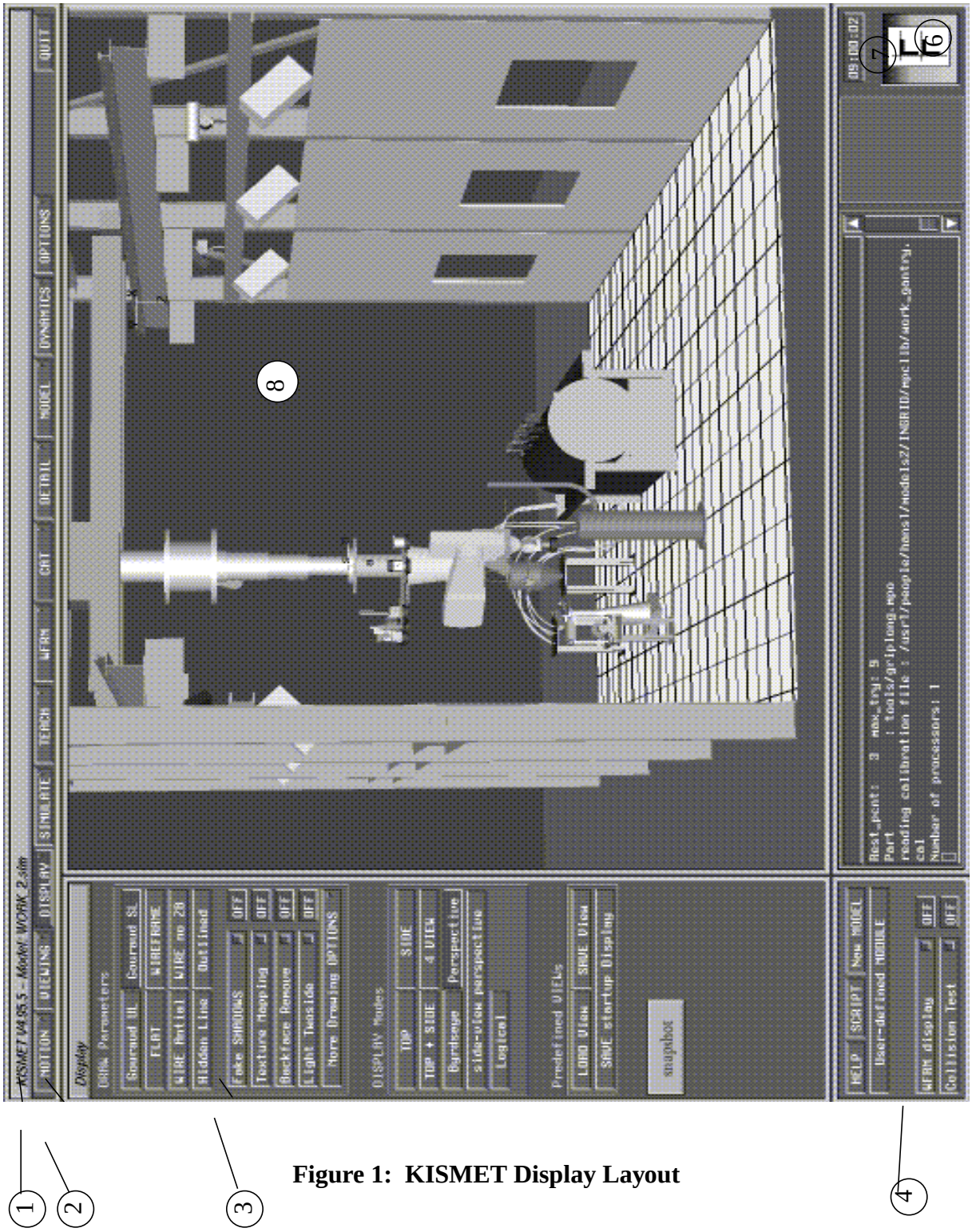

Figure 1: KISMET Display Layout

The workcell viewport (8) covering an area of 1000x800 pixels is surrounded by 7 fields described in figure 1. Using KISMET, up to 4 different scenic views of the workcell can be dis-

played simultaneously. In the so called „full screen mode“ the whole screen is used for workcell display.

1	Displays the current KISMET version and the loaded model.
2	Main Menu selection panel (In the displayed example the submenu DISPLAY has been selected). All menus of the Main Menu are described in detail in chapter 6 List of Commands to chapter 17 OPTIONS.
3	Sub menu selection panel. Depends on the selected submenu.
4	Function selection panel. See chapter 18 Function selection panel for details.
5	Textport: Used for the input of alphanumeric data (e.g. filenames etc) and the output of certain messages.
6	Logo of Forschungszentrum Karlsruhe
7	Diplay of actual system time.
8	Viewport.

Table 1: KISMET Display Layout

4 Program and User Environment

4.1. KISMET Module Structure

The KISMET simulation environment is a combination of program modules. These are:

KISMET this is the KISMET system main module. It has all features for model creation, editing, simple graphical robot program teaching in IRDATA, and simulation. It consists of two programs, the **'kismet_kern'** and **'kismet'** processes, which are the simulation kernel (responsible for workcell drawing) and the man-machine interface (MMI), respectively. The 'kismet_kern' process can be run without the MMI process 'kismet'. If both are executed together (the "normal" mode), 'kismet_kern' is called by 'kismet' after program start.

2Ded is a simple 2D-editor, which is used by 'kismet_kern' during 3D-geometry modelling with some specific modelling primitives (sweeps). Its operation is described in chapter 21 The 2Ded-Editor for SWEEP Operations

GRIPS is a textual off-line programming system for KISMET using the PASRO high-level robot programming language, and an IRDATA interface to KISMET for program simulation and testing. A description of GRIPS is given in [3]. GRIPS itself consists of a few submodules:

- **wsh_GRIPS**

  command script to start GRIPS from kismet as an overlay. GRIPS is forked as a parallel process to kismet.

- **GRIPS**

  is the main module for high-level, textual robot programming with KISMET. It starts and controls the other programs listed below. It communicates with kismet for program simulation.

- **PASIR**

  is the PASRO to IRDATA compiler.

- **wsh_shell**

  command script to fork the Edit module as a parallel process.

- **Edit**

  is a filter program for PASRO-program editing. It is using the UNIX system 'vi'-editor.

create_dir: syntax: create_dir <path>

create_dir is a short and simple command script to create a all directories of KISMET model-library. It creates the subdirectories compact.

- <path>/envlib
- <path>/mpclib
- <path>/mpolib
- <path>/teachlib
- <path>/scripts

Note:

the directory denoted by <path> must already exit prior to calling *create_dir*. For a detailed description of the file and directory system used by KISMET see 4.2. FILETYPE and DIRECTORY-STRUCTURE Overview

diagram is a program module for multi-diagram display. The program is using position time-series data generated from kismet during robot-program execution to display diagrams (joint and TCP-position/ -velocity/ -accelleration over time). It is capable of multi diagram display.

Figure 2 shows a simulated robot motion, displaying the position, velocity, acceleration and torques.

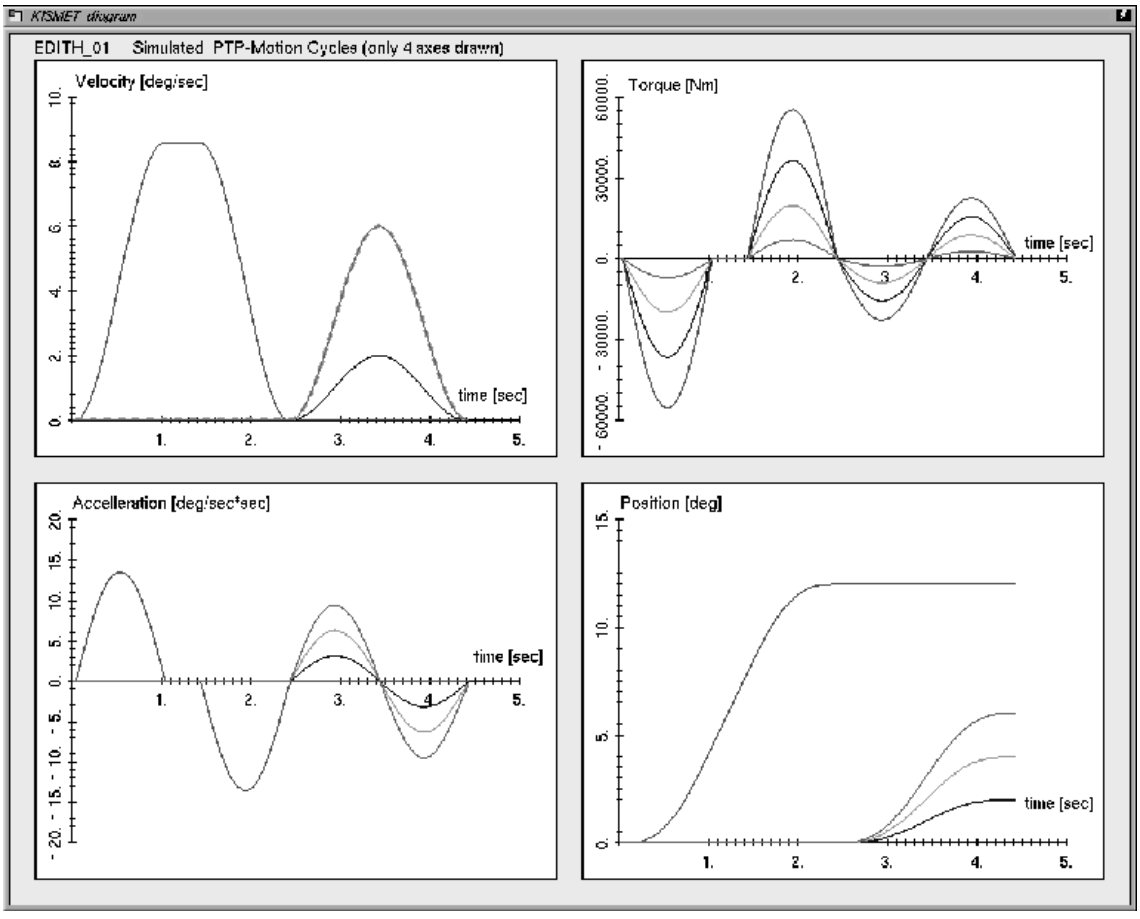

Figure 2: Display of Position, velocity, acceleration and torques during robot movement

4.2. FILETYPE and DIRECTORY-STRUCTURE Overview

All datafiles used in KISMET use a textual ASCII format and may be entered by means of text editing (on UNIX-systems 'vi', 'emacs') or through the interactive graphical editing options of KISMET. This chapter gives an overview of the different filetypes used by KISMET for an application model.

Usage of textual file formats also simplifies the development of CAD-processors for the transfer of model-data from commercial CAD systems into the KISMET-specific dataformat. The CADI postprocessor KISPOST, (covering geometry only) and a STEP postprocessor (covering geometry and kinematics) were developed and are available for KISMET.

4.2.1 Introduction to KISMET-Filetypes

The user specifies during the initialization the filename of the Simulation CONTROL-File (file-extension '.sim'). This file contains references to all datasets required by KISMET for a specific application.

The datasets specified in the '.sim' file define these characteristica of a simulation run:

2. The Viewing LAYOUT-File (file extension '.lay') defines the initial KISMET window-layout, i.e. position and size of the different scene-views (viewports) and the viewing-parameters of each view (the initial viewing position, the view-direction, size of the displayed 3D-world window), and the mode of projection to be used for each scene view (perspective or orthogonal projection).
3. The LIGHTING-File (file extension '.lgt') allows the definition of up to 8 parallel- or point-lightsources.
4. The MATERIAL-File (file extension '.mat') specifies the surface properties required for diffuse and specular reflection which is calculated for shaded-rendering. The user can define for each application the different components of the lighting effects on the surface of model parts for various materials.
5. The COLOUR-File (file extension '.col') defines two colour tables for KISMET. The first one is a table of constant colours (no influence of lightsources), which is used for the wireframe-rendering or texturing, text etc. on shaded objects. The second table defines the basic colour for the different MATERIAL definitions used for shaded rendering.
6. The ABSTRACT-BASE File (file extension '.mpc') defines the root and uppermost hierarchical level of the model-tree. All other model parts and branches are referenced from within this dataset. 'ABSTRACT-files' (or COMPONENT-files) may define in every 'FRAME'-node another 'ABSTRACT-File' on a more detailed hierarchical level. Theoretically, indefinitely deep detail-leveling and hierarchies can be realized.

ABSTRACT-Files define

- Moveable mechanical structures like industrial robots, handling manipulators and tools, gearings and other machine elements.
- Working environments (workcells).
- The ABSTRACT file ('.mpc') may include references to the following filetypes:
    - GEOMETRY-files (extensions '.mpo', '.MPO'). For each GEO-element, this is for each ENTITY, a single GEO-file is used. GEO-elements are linked to a FRAME-reference through a PLACEMENT transformation.
    - DOF-files (extension '.dof'). These files are mandatory references in ROBOT-

ABSTRACT files only. They define input/mechanism funcional relations. Each INPUT-Function may act on one or multiple mechanism degrees-of-freedom (DOF).

- CALIBRATION-files (extension '.cal'). These files are mandatory references in ROBOT-ABSTRACT files, if the ROBOT is to be in monitoring applications. The file contents define interface parameters for the joint-position interface to the real robots controller.

4.2.2 Structure of a PROJECT DIRECTORY

KISMET supports the concept of project directories. This means that the user can open a new model directory for every new application and/or project. A project directory contains the KISMET specific subdirectories.

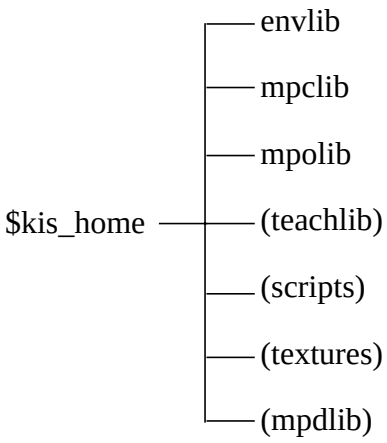

Figure 3: KISMET Directory structure

During program execution, KISMET is accessing different datasets in subdirectories which are predefined for the different filetypes. For example to read a robot program with the filename 'ROB_prog1', KISMET completes the searchpath from the project directory with 'teachlib'. The complete UNIX pathname is therefore:

$kis_home/teachlib/ROB_prog1

A robot program which was offline programmed in KISMET is automatically saved (written) into the subdirectory 'teachlib'.

The term '*$kis_home*' is a UNIX shell variable, which defines the searchpath for the project directory. This variable is to be set by the user according to the project directory in the command shell process environment using the csh-command setenv to allow KISMET access to the simulation model datasets.

To set the shell-variable 'kis_home', add the commandline:

setenv kis_home /usr/people/kis_applications/project_id

into the user login file '.cshrc'.

The filename of the robot program 'ROB_prog1' in the previous example thus would be expanded by KISMET to the complete UNIX pathname:

/usr/people/kis_applications/project_id/teachlib/ROB_prog1

The subdirectories for the various filetypes used by KISMET cannot be respecified by the user and are predefined as follows:

envlib The 'ENVIRONMENT-LIBRARY' hosts files of the types '.sim' (SIMULATION CONTROL-Files), '.mat' (MATERIAL definitions), '.col' (COLOUR-Files), '.lay' (Viewing LAYOUT-Files), '.lgt' (LIGHTING-Files), '.klt' (deines Texture lookuptables and transfer functions) and '.dof' (defines input degrees-of-freedom and functionally defined mechanisms).

mpclib The 'CONFIGURATION-LIBRARY' stores 'ABSTRACT'-files (COMPONENTS) of type '.mpc'. These datasets are used for the definition of the topology of assembly structures and kinematically defined mechanisms.
Additionally datasets of type '.cal' are stored in mpclib, which are used to calibrate the sensor interface used for robot monitoring.

mpolib The 'OBJECT-LIBRARY' serves for storage of GEOMETRY and binary Volume datasets. For any defined GEOMETRY entity, a separate file with the extension '.mpo' is required in KISMET.

teachlib Robot programs are stored in the TEACH-LIBRARY 'teachlib'. Because KISMET was initially designed for the simulation of different robot programming languages, there is no suffix predefined or checked for robot programs. Special requirements for the filenames are often caused by the robot controllers. The naming of robot programs therefore is totally up to the personal taste of the user. By convention the suffix '.ird' is used for IRDATA programs.

Additionally, WORKFRAME PATH-files ('.wpf') are stored in this directory. These files provide the datalink between the geometrical database of KISMET and high-level robot programs. These files are used by external Off-Line Programming modules (OLP) prior linkage of executable IRDATA-files.

scripts The 'SCRIPT-LIBRARY' stores datasets of type '.spt' (SCRIPT-Commandfiles) and '.vws' (binary VIEWING-Files). These files are used to save a specific viewing configuration of the KISMET screen layout.

textures The 'TEXTURE-LIBRARY' stores SGI image-files in RLE-format, which are used as texture definitions for the texture-mapping rendering capabilities of KISMET. There is neither a special suffix predefined nor checked for texture files. Texture files are referred to from MATERIAL-files ('.mat').

mpdlib the 'DYNAMICS-LIBARY' 'mpdlib' hosts files of the types '.def' (DYNAMIC DEFINITION-Files), '.stf' (STIFFNESS-Files), '.ita' (INERTIA-Files), '.diag' (DIAGNOSE-Files), '.con' (SIMPLE-CONTROL-Files), '.con' (SIMPLE-CONTROL-Files), '.csf' (COMPLEX-CONTROL-Files), which are usefull for the 'DYNAMICS', 'CONTROL' and 'ELASTOMECHANICS' features.

4.3. UNIX Environment Variables

Prior to start KISMET for the first time, some variables have to be set in the user environment

of your UNIX command shell:

kis_home	this environment variable contains the searchpath for the model-data of your current KISMET application.
kis_help	gives the directory (path) for KISMET helpfiles
PASRO_defs	stores the searchpath for the modules used by the textual robot programming module GRIPS, which can be started from KISMET. The program- and datafiles PASIR (PASRO to IRDATA compiler), wsh_shell (command script to start the 'vi' editor) and RAHMEN.pas (empty PASRO-program sourcecode body) are started relative to the path defined by PASRO_defs.
zem_pd	defines the path for the data- and configuration files of the diagram module.

These variables should be set to a default directory. To do this, enter similar lines into the file .login in your account, as follows:

```
setenv kis_home /usr/people/my_directory/kis_model1
setenv kis_help /usr/local/kismet/help
setenv zem_pd /usr/people/my_directory/kis_model1/diag_data
setenv PASRO_defs /usr/people/my_directory/bin
```

5 The KISMET User Interface

KISMET can be operated in two different modes:

1. The default **panel-MMI mode** using the DISPLAY menus and buttons as described for example in Chapter 3 KISMET Display
2. The **full screen mode** where KISMET is operated to a large extent using popup menus. According to their function, the individual commands are organized in a menu tree.Using the MOUSE, which is provided as a standard input device with your graphics workstation, the individual commands are selected from the popup menus. See chapter 20 The Full Screen Mode for details.

Alphanumerical data (e.g. filenames) are entered using the keyboard. For some operations, FUNCTION KEYS are provided.

5.1. Program Startup and Commandline Options

KISMET is installed in the UNIX-directory */usr/local/kismet/bin*. It can be started by any user whose search path contains this directory. If you have problems to start KISMET, check if it is installed in */usr/local/kismet/bin*, or if this directory is included in your path.

SYNOPSIS:

kismet [SIM_filename] [options]

Options:
-d
-a <model_accuracy>
-s [1...20]
-v170
-pal
-l
-S
-nil
-e
-dgl
-pk
-up
-w
-gt
-NOVOLREN
-4Bit
-cf <control_filename>

DESCRIPTION:

The filename of the simulation control file <SIM_filename> can be defined optionally in the command line. Otherwise this parameter is requested from the user after KISMET has been started in the file-selection box.

The option **-d** starts KISMET in diagnostic mode. This can be used to display additional information in the textport while all the model data files are read by KISMET.

Using the **-a** option, a minimum modelling accuracy can be defined in 'mm' for geometry parts which require a large number of computations during display, such as isometric definitions of pipes (PIPE primitive) and rotational solids (ROTATIONAL_SWEEP). The smaller the given parameter value <model_accuracy>, the higher is the number of generated surface polygons (facets) generated by KISMET during model startup and, hence, the smaller is the frame rate, i.e. the number of calculated views per second. The default value for <model_accuracy> is 1.0mm.

The option **-s** defines the number of subdivisions in each direction per segment patch for polynomial free-form surfaces according to VDAFS 1.0 (CURVE- und SURF-Primitiv), which are used by KISMET to approximate the geometry surface.

For video output purposes (typically using the SGI multi-channel option and/or for display using a video beamer, or a VR-HMD), 2 more display options are available to create the KISMET workcell window with predefined sizes. The size of the KISMET window corresponds to the video signal of the workstations video output channel which may be connected to a specific projection device:

- the **-vt170** option is used for NTSC video output while
- the **-pal** option is used for PAL video output

Using the latter options also implies full-screen mode. The panel-MMI is not started.

For simulations with extended lighting-models (KISMET version V4.0 and higher), i.e. with SPOTLIGHTS and lightsources with defineable RGB-colour, you should start KISMET using the **-l** option. This activates the extended MATERIAL definitions which are required for more realistic display.

The **-S** option is used to catch fatal signals. It is mainly useful for the KISMET development team to create core files.

The **-nil** option forces KISMET to skip the reading of so called inner loops. These inner loops are for example used to describe holes in a geometric object.

To use KISMET together with the EVE projection dome, use the **-e** option.

If you run KISMET on a multiprocessor Silicon Graphics workstation KISMET is forced to run certain kinematic calculations on the second processor by using the **-pk** option.

If you are running KISMET on a multiprocessor *Silicon Graphics* workstation you can force KISMET to use only one processor, if you start KISMET in the single processor mode by using the **-up** option (uniprocessor-mode). By default, KISMET is using 2 CPU’s on multiprocessor workstations. One for graphics and the second for realtime ’Elastodynamics’ and other computing-intensive calculations (apart from graphics), i.e. the second processor is the "number-cruncher". If the current KISMET model is not doing any ’Elastodynamics’, the allocation of the second processor is useless.

If you want to run KISMET in the full screen mode inside a window use the **-w** option. If you use the **-wb** option a frameless window will be used.

With the **-dgl** option it is possible to run KISMET on one workstation while the output is done on a second workstation defined by the DISPLAY variable.

If you want to switch off GEO-test during model startup, use the **-gt** option.

The -**NOVOLREN** option:
Since version V 6.0, you may render voxel-based volume-models in KISMET (volume-rendering). This graphics feature requires specific realtime graphics hardware capabilities. The

"GL_TEXTURE_3D_EXT" extension to OpenGL is used by KISMET for this purpose. Now, the **-NOVOLREN** option is used to prevent initialization of the volume-rendering hardware during KISMET startup. We had some program crashes when running KISMET across the network with the DISPLAY on an IRIX 5.3 workstation without 3D-texture capabilities. This option is a workaround of the problem.

The **-4Bit** option:
Use 4-Bit Textures for volume rendering (the default is 8-Bit 3D-textures for each colour channel). On some graphics workstations, this option may speed up volume-rendering with volume-models larger than the texture-memory size. Since the 3D-texture bricks have to be downloaded to the texture-memory for each rendering pass (frame), using 4-Bit textures halves the amount of data to be downloaded. Thus, the purpose of this option is to increase rendering speed for a sacrifice in image quality.

The **-cf** option:
Start KISMET with a '.kctl' control file (defines both, the 'kis_home' environment variable and the '.sim' filename). This option is required, when KISMET is started from the *Netscape* browser with a specific model.

KISMET Startup Examples:

```
kismet zelle01a.sim -d -a 2.0 -up
```

The simulation file 'zelle01a.sim' is started in diagnostic mode. The modelling accuracy will be equal or better than 2.0mm and a single-processor computing architecture is used on multiprocessor workstations.

```
kismet catrob2.sim -d > diag.data
```

The simulation file 'catrob2.sim' is executed in diagnostics mode. The text output during model startup is routed into the file 'diag.data' (rerouting of the standard output to a file in UNIX).

5.2. Logical Input Devices

In the description of individual commands in the following sections, the mouse buttons are reffered to as follows:

LM	left mouse button
MM	middle mouse button
RM	right mouse button
BM	the left and middle mouse buttons pressed together, i.e. to execute the function, both buttons have to be pressed simultaneously.

The logical input devices required by the different KISMET functions can be classified as follows:

STRING	denotes an input device generating an alphanumerical string as a result of the input function. For this, the keyboard of the workstation is used.

LOCATOR is used to specify model-, world-, or screen-coordinates. Input is carried out using the mouse together with the screen cursor. Important for the function is the position of the cursor on the screen and not that of the mouse on the tablet.

Depending on the functions, either an arrow cursor or a crosshair cursor (two crossed lines on the screen) are used. By shifting the mouse on the tablet, the cursor position on the screen is changed. General handling of the mouse is explained in the manuals of your SILICON GRAPHICS workstation.

VALUATOR is used for scalar input data. Generally, interactive input of valuator data in KISMET is carried out using the mouse. The scalar value (e.g. joint position, display zoom factor etc.) is varied incrementally in accordance to the differential position of the mouse.

Usually, only one of both (X and Y) input channels is used (X-channel) to evaluate the VALUATOR input. To release the input function, one or several of the buttons (LM, MM or BM) have to be pressed together with the mouse motion. Thus, up to three different functions can be carried out alternatively in a VALUATOR function mode. For example, in the VALUATOR function mode "Change view direction", LM is used to change the azimuth view angle, MM for the elevation angle, and BM (pressing LM and MM together) zooms the display (change of the viewing angle).

The right mouse button (RM) is reserved in KISMET for the popup menus. It is not used in any of the VALUATOR functions. Accidental movement of the mouse without simultaneously pressing one of the mouse buttons does not have any adverse or side-effects on KISMET.

CHOICE selects the menu options and toggle functions. Toggle functions are those operations which offer a TRUE/FALSE, YES/NO or ON/OFF choice.
In principle, the popup menus are brought up on the screen by pressing the right mouse button (RM), i.e. the next menu up in the menu tree is selected and displayed on the screen. A function selection in a menu is made using LM or MM. When LM or MM are clicked and the cursor is outside the menu, no selection is performed and the menu disappears. If RM is pressed, the next menu level is displayed (nearer to the main menu).

PICK is used for the selection of model parts. A typical example of a PICK operation is the selection of a GEOMETRY element, which is to be geometrically transformated (rotation, translation) using the following VALUATOR function. To perform a PICK operation, move the cursor over the object and press LM or MM. Picking is stopped and cancelled (no selection) when RM is pressed.

To PICK a GEOMETRY element, move the cursor over the screen area covered by the part, or better, over the PICKING-region (the little white rectangle displayed approximately in the middle of the part). Geometry picking is performed faster when the part is picked over the PICK-region.

To pick other objects, like FRAMEs (i.e. hierarchical model part reference coordinate systems), ABSTRACTs (hierarchical assembly or detailing-level structures of the model, which are grouped in a '.mpc' definition file), or WORKFRAMES (tool reference position on the objects to be handled), move the cursor to the origin of the coordinate system and press LM or MM. For FRAMEs, ABSTRACTs or WORKFRAMEs (WFRM), little PICK-regions are displayed too, at the origin of the coordinate frame (in PICK-mode only).

5.3. Screen Mascs in KISMET

5.3.1 Utilization of the File-Selection-Menu

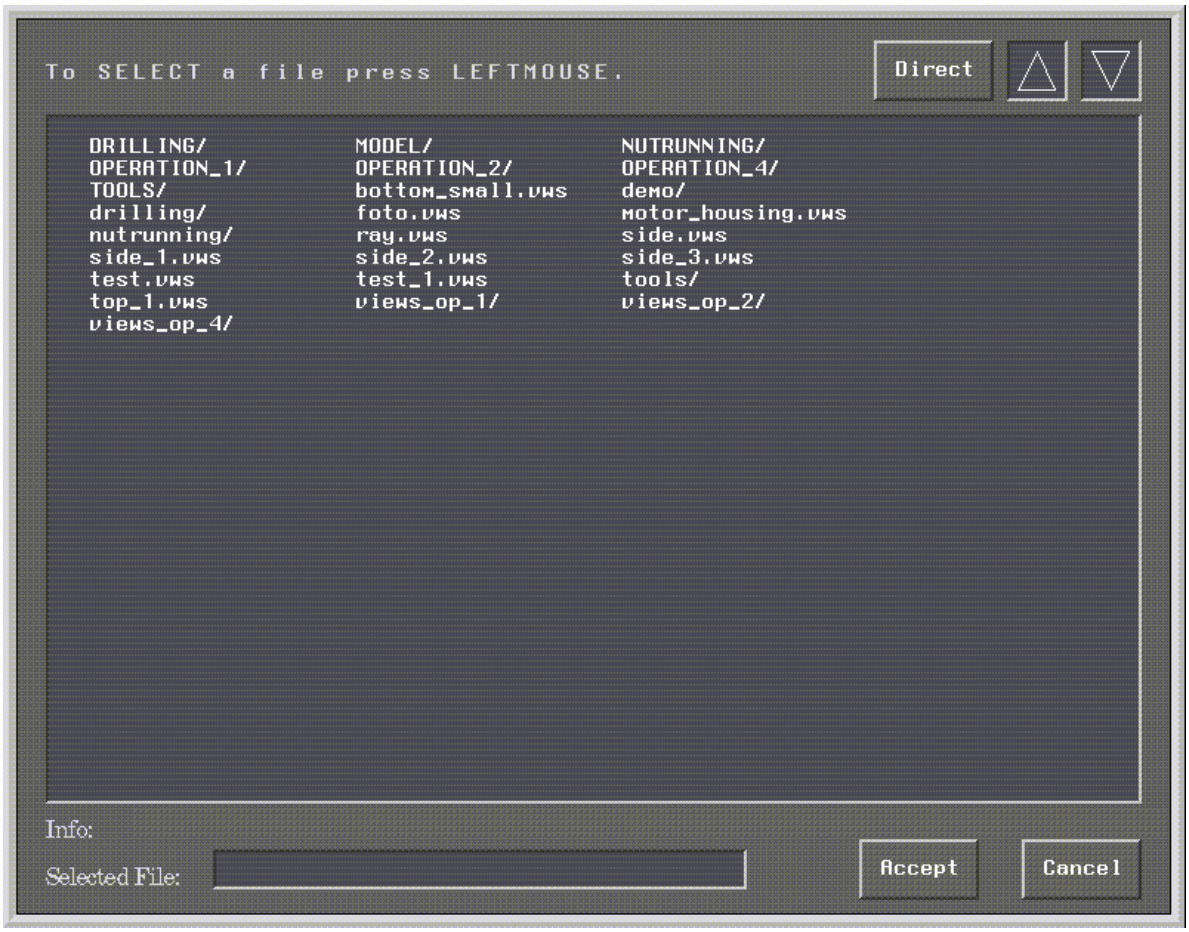

Figure 4: Example of a file-selection-menu

When the file-selection-menu is started, a list of files is shown in the opened window. You can select a file when the mouse is above its name and you press *leftmouse (LM)*. The filename appears in the *selected-file-field* at the left bottom of the window. In the *info-line*, one can read a comment about the file which is currently selected by the actual mouse-position.

Directories are marked with a slash at the end of their names. If you select a directory name by pressing leftmouse, the actual path will be changed. The files which are contained in the new current subdirectory appear on the screen. To go back to the superior directory one must select the *Direct-button* with the mouse and press leftmouse (LM) to activate it.

The buttons with the triangle in the right top of the file-selection-menu-window have the function of scrolling up and down the list of filenames. They become activated by pressing leftmouse (LM) as long as you like to scroll.

When the mouse is above the *Accept-button* and you press LM the selected filename is passed to KISMET for further processing and the menu is going to be left.

If you activate the *Cancel-button* the menu will be left without loading the selected filename.

Note: When the mouse is above a button, this button is active and can be used. To make this visible, the accept-, the cancel- or the direct-button are printed in VIOLET, the triangles of the

scroll-buttons are printed filled out in GREEN and a filename is written in RED instead of BLACK.

5.3.2 Utilization of Input-Forms

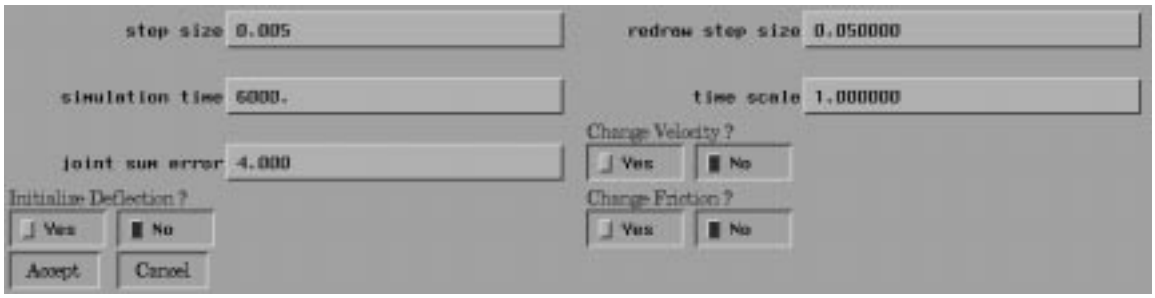

Figure 5: Example of an Input-form

An input-form consists of i*nput-field(s)* and/or l*ine(s) of buttons*. One input-field or one button is selected for input when the mousepointer is over it. The following input-modes exist:

input-field: In the left of a field there is a short descriptive comment. A default value is displayed in each input field. It can be erased by using the *backspace-key.* With other keys you can modify the string.

button-line: When the mouse is above a button it is active and can be set by pressing leftmouse. One of the buttons of each line is set by default. The small rectangle which is displayed on each button indicates the current function status (white colour: button is not set;violet colour: button is set). Above each button-line and upon each button there may be printed comments respectively.

Accept-button: When the mouse is on this button and you press leftmouse all modifications you have made will be saved and the input-form will be left.

Cancel-button: In this case the input-form will be left without saving any modification.

Note: When an input-field or a button is active, i.e. the mousepointer is above it, the input-field is marked by writing the string in MAGENTA instead of GREEN, the button is printed in DARK_GREY instead of BRIGHT_GREY.

6 List of Commands

6.1. Introduction

This chapter describes the KISMET menues visible for the user while executing KISMET in the non-full screen mode. In full-screen mode a different user interface using pop-up menus will appear. See chapter 20 The Full Screen Mode for detatils.

6.2. Main Menu selection panel

The following commans are provided by the KISMET Main Menu:

Figure 6: The Main Menu Panel

Each selection of one of the buttons opens a panel on the left hand side of the screen (see chapter3 KISMET Display):

- **MOTION**

  Leads to the MOTION panel (see chapter 7 Motion-Submenu). It is used for mechanism motion control. Motion commands always move the active ROBOT.

  Note: That mechanism model (kinematical structure, ROBOT) with current input focus for the motion valuator commands, is called "*active ROBOT*" in this document.

- **VIEWING**

  Selection of the VIEWING-submenu. This leads to a group of commands, which are used to vary different parameters of the workcell views (viewing position, view direction, zooming factor, size and position of the viewports on the graphics screen etc.) See chapter 8 VIEWING.

- **DISPLAY**

  Leads to the DISPLAY-submneu for selection of different screen layout options. See chapter 9 DISPLAY

- **SIMULATE**

  Leads to the Simulate submenu. This menu is used for simulated robot program execution. The submenu includes the commands for robot program load, start, stop, verify and delete. KISMET provides a virtual robot controller for each robot (mechanism) defined in the model datafiles. The virtual controller stores the parameters and variables of the robot. This is, for example current speed, accelleration for the TCP (linear movements) and joints (PTP move) TCP- and ZP-offsets, pointers to loaded robot programs etc.. See chapter 10 SIMULATE.

- **TEACH**

  Leads to the TEACH submenu. This menu is used for robot program generation using simple geometric teaching. This is, the robot is moved to the next target position in KISMET. When the user is satisfied with the position, the robot motion command can be generated for this position, followed by defining the next position and so on. See chapter 11 TEACH.

- **WFRM**

  Leads to the WFRM-panel for creation, editing and positioning of "Workframes" (WFRM).

A WFRM is the supertype datastructure which is used in KISMET to define TCP reference positions on the objects to be handled. The basic task in robot programming is to move the robot TCP in a specified way from one WFRM to the next and perform at these positions a specific task. To automate or simplify off-line program generation, KISMET allows the definition of additional data with each workframe. This is a name (identifier), a comment, a type-id, a visibility index etc.. The WFRM is connected to a FRAME through a transformation matrix, which allows local, relative positioning. A group of WFRMs in a specific, user defined order, forms a WFRM-PATH. A set of WFRM-PATH structures is stored in a WFRM-PATH file.

WFRM-data can be passed during KISMET program execution to other programs through a file. A SCRIPT command is used to initiate this action.

Additionally to the standard function of a WFRM as described above, the WFRM datastructure is used in KISMET to define CAMERAs, moveable scene LIGHTS, USS-sensors (simulated ultrasonic sensors), robot TCP- and ZP-systems.

See chapter 12 WFRM.

- **CAT**

  Leads to the CAT (Computer Aided Teleoperation) panel. See chapter 13 CAT.

- **DETAIL**

  Leads to the DETAIL-submenu, which is used to load, activate and deactivate model parts (ABSTRACTS) and details as well as to create new models.

  In KISMET, an ABSTRACT is a kind of a special FRAME, which acts as an reference for an assembly, or to another, more detailed representation of the part. The physical representation of the ABSTRACT is a file (’mpc’-file) which defines a part with it’s FRAMEs and file-references to the GEO-element data (’mpo’-file). An ABSTRACT is also used to define exactly one ROBOT.

  As a special feature in KISMET, each FRAME defined in an ABSTRACT-file can store the file-reference to another ABSTRACT-file. This feature is used to define multiple levels of detail in KISMET. See chapter 14 DETAIL.

- **MODEL**

  Leads to the Model submenu containing submenus responsible for

  - editing and changing of geometric parts (GEO-parts),
  - creating and moving of FRAMES,
  - editing colour tables,
  - editing tables defining surface properties, so called "MATERIALS".

  See chapter 15 MODEL.

- **DYNAMICS**

  Leads to the DYNAMICS-submenu for real time simulation of the dynamic behaviour of mechansims as well as robots and their control systems. See chapter 16 DYNAMICS.

- **OPTIONS**

  This menu collects a set of commands and submenus which do not fit in one of the other

group and/or areused less frequently. See chapter 17 OPTIONS.

- **Quit**

  is used to terminate the execution of KISMET. You should always use this command to exit KISMET, i.e. you should not interrupt it. Otherwise the state of the graphics hardware may be unpredictable (stereo, video framegrabber options). KISMET will initiate all used devices to a reasonable default state just before the exit. Additionally, KISMET is opening shared memory channels for interprocess communications. Shared memory is not automatically closed, when the program is stopped. These are the reasons for always using "Quit" to exit.

6.3. The function selection panel

The function selection panel is described in detail in chapter 18 Function selection panel.

7 Motion-Submenu

7.1. Introduction

The "Motion"-submenu is used for mechanism motion control. Motion commands always move the active ROBOT. For motion commands we distinguish in KISMET two classes of mechanisms:

DEVICE A DEVICE is a mechanism with any number of JOINTs or degrees-of-freedom, but with no inverse solution available in KISMET. Thus, a DEVICE cannot move in tool-coordinates, because this requires the inverse kinematics. To move a DEVICE however, the motion commands in JOINT space („Single-Joint Control“, „On/Off speed control“, „Enter JOINT values“) are available.

ROBOT A ROBOT is a kinematical structure with available inverse kinematics solution. This is, it can be moved in tool-coordinates. A tool-center-point (TCP) system must be defined in the kinematics definition file (‚.mpc‘ file) of the robot. All motions in tool coordinates are carried out with respect to this TCP-system. To define interactively a TCP, use the "WFRM"-menu (see chapter 12 WFRM). To apply an offset to the TCP, use the "TCPF Setup"-command (see chapter 7.2.1 The TCPF-Setup Panel)

In KISMET, two ways of inverse kinematics solutions are implemented:

- An analytical solution is used for some classes of 5- and 6-axis industrial robots, and for the JET „Articulated Boom“ and FZK „EDITH“ RH-transporters. The analytical solution chosen according to the ROBOTs kinematical class identifier in the ‚.mpc‘ definition file. Internally, one subroutine is implemented for each of these classes. The analytical solution is always the fastest way to solve the inverse problem. It is only possible if certain kinematic conditions are fullfilled.
- A numerical algorithm for the inverse problem is implemented for those robots with no analytical solution. The method used is called Modified Newton-Raphson (MNR) algorithm. In fact, 3 cases are solved in KISMET:
    - ROBOTS with 6-DOF, or more precise, the DOFs exactly fullfill the conditions to reach an arbitrary position and orientation in its 3D working space.

- Robots with less than 6-DOFs, or more precisely, the robot cannot reach any arbitrary position and orientation. The problem is solved for a subset of the 6-DOFs in 3D space.
- ROBOTS with more than 6-DOFS, i.e. with kinematic redundance. Because of allocated memory, the current limit of JOINTs in KISMET is 10. This number may be increased in future versions, if required.

In all 3 cases the Jacobian is calculated numerically, in the first case the solution is found by matrix inversion. For the latter two cases, the pseudoinverse solution of the Jacobian is used (Moore-Penrose Inverse). The processing time for one iteration in KISMET is in the order of 1ms. For small movements (within ‚mm') usually 1-2 iterations are sufficient. Larger movements require up to 10, mostly less than 6 iterations. The target position is reached with an accuracy of better than 0.002 mm (stop condition). Most of the motion commands are VALUATOR functions. The input focus is attached to the active ROBOT. To move another ROBOT, use the „Activate ROBOT" command (in the „main-menu") first.

7.2. Motion: List of Commands

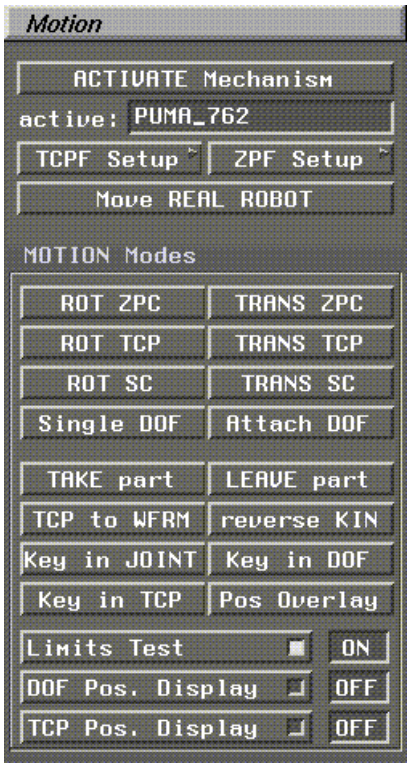

Figure 7: The Motion Panel

- **Activate Mechanism:**

  Selection of the active mechanism which can be a robot or any other kinematic structure. The selected mechanism is displayed in the *active-field*. All robot specific input/output commands, i. e. motion control commands, definition of TCP- and reference-system displacement, loading of robot programs are carried out in relation to the active mechanism.

- **TCPF Setup:**

Leads to the ***T**ool-**C**enter-**P**oint-**F**rame Setup* menu which defines the positioning of the robot Tool-Center-Frame. See chapter *7.2.1 The TCPF-Setup Panel* for details.

- **ZPF Setup:**

  Leads to the ***Z**ero-**P**oint-**F**rame Setup* menu. See chapter 7.2.2 The ZPF Setup Panel for details.

- **Move REAL ROBOT:**

  Can be used to control a real robot by sending position commands to the robot control system.

- **ROT ZPC:**

  Valuator function. Rotates the robot TCP-system in cartesian ZP-coordinate space (inverse kinematics required). A rotation of the global orientation angles A,B,C (RPY) is carried out.

  - **LM:** Change A-orientation
  - **MM:** Change B-orientation
  - **BM:** Change C-orientation

- **TRANS ZPC:**

  Valuator function. Shifts the robot TCP-system in cartesian ZP-coordinate space (inverse kinematics required). The orientation of the TCP-system is not affected by this command.

  - **LM:** Move in X-direction of the ***Z**ero-**P**oint-**C**oordinate system*
  - **MM:** Move in Y-direction of the ***Z**ero-**P**oint-**C**oordinate system*
  - **BM:** Move in Z-direction of the ***Z**ero-**P**oint-coordinate system*

- **ROT TCP:**

  Valuator Function. Movement in tool-coordinates. Rotates the robot TCP-system around local TCP-axes (inverse kinematics required). On the scene display, the local TCP-axes are named n,s,a (normal, side, approach). This naming corresponds to x,y,z directions. The position of the TCP-system is not affected by this command.

  - **LM:** Rotate around **n**-axis (X)
  - **MM:** Rotate around **s**-axis (Y)
  - **BM:** Rotate around **a**-axis (Z)

- **TRANS TCP:**

  Valuator function. Movement in tool-coordinates. Shifts the robot TCP-system parallel to the current tool-coordinate directions (inverse kinematics required). The orientation of the TCP-system is not affected by this command.

  - **LM:** Shift along **n**-axis (X)

- **MM:** Shift along **s**-axis (Y)
- **BM:** Shift along **a**-axis (Z)

- **ROT SC:**

  Valuator function. Rotates the robot TCP-system in cartesian screen coordinates (inverse kinematics required).

- **TRANS SC:**

  Valuator function. Shifts the robot TCP-system in cartesian screen coordinates (inverse kinematics required).

- **SINGLE DOF**

  Is used to move up to two DOFs in joint space. For the valuator operation you can attach LM or MM of the mouse each for a different joint by using the following **ATTACH DOF** button.

- **ATTACH DOF:**

  Used to attach LM and/or MM of the mouse for different joints if **SINGLE DOF** has been chosen. On screen the joint attached to LM is drawn in BLUE, the one attached to MM is drawn in RED. Each time you select the ATTACH DOF button, the „JOINT“-menu is displayed. To attach the valuator input channel of LM to a joint, press this button over the joints name in the „JOINT“-menu. To attach MM, proceed accordingly.

- **TAKE part:**

  Tells the robot to take a part. The data tree is searched for workframes (WFRMs). That WFRM with the closest distance to the robots TCP is identified and the corresponding FRAME (MPKIN) is connected to the robot hand. Only those WFRMs are identified, which are in range (constant MIN_TCP_WFRM_DIST) of the TCP are allowed. General information concerning WFRMs can be found in chapter 12 WFRM.

- **Leave part:**

  Tells the robot to leave a part which has been connected to the robot hand using the **TAKE part** command.

- **TCP to WFRM:**

  Moves the ROBOT TCP to the location of a workframe (WFRM). The WFRM target position is defined by means of a PICK operation, after the command was triggered. This function requires the inverse kinematics solution. Depending on the kinematics structure of the robot and the definition of the kinematic behaviour, it may be possible, that the target position is reached only partly. For example, a specific robot may only allow alignment of the position and the X-component of the orientation frame.

  If the position (location and orientation) can for some reason not be reached (out of range, joint limits etc.), an error message is given.

- **reverse KIN:**

  Reverses the kinematic structure of the robot. The robot base (the Zero-Point Frame) beco-

mes the robot TCP and vice versa.

- **Key in JOINT:**

  Alphanumeric input of a robot position. KISMET requests the value for each joint in the textport.

- **KEY in DOF:**

  Alphanumeric input of a robot position. KISMET requests the value for each degree-of-freedom in the textport.

- **KEY in TCP:**

  Alphanumeric input of the robot position. KISMET requests the orientation and position of the robot TCP frame with reference to the robot base. (Inverse kinematics required).

- **Pos Overlay:**

  The actual position of the robot will be shown as alphanumerical values in the graphical viewport as an overlay display.

- **Limits Test ON/OFF:**

  Determines whether the joint limits will be checked or not.

- **DOF Pos. Display ON/OFF**

  The current position of the active robot will be displayed. The display can be used in order to change the position by changing the displayed values.

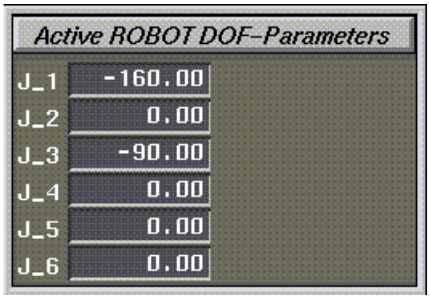

Figure 8: The Active ROBOT DOF Parameters:

- **TCP Pos. Display ON/OFF:**

  The current position and orientation of the Tool Centre Point will be displayed. The display can be used to change the position/orientation by changing the displayed values.

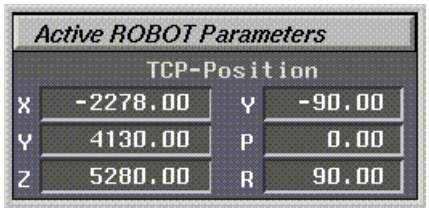

Figure 9: TCP-Position display

7.2.1 The TCPF-Setup Panel

This is the submenu for definition of the active robots tool-center-point (TCP) offset parameters. The TCP-offset is the reference system for robot motion commands. It should be placed into the work position of the current tool. If the TCP-frame is displayed in KISMET, the orto-normal TCP-orientation vectors are defined through:

n: normal vector (X-direction)

s: side vector (Y-direction)

a: approach vector (Z-direction).

For all robot program motion commands, any target position may be defined in JOINT-SPACE or in cartesian TCP-SPACE. In cartesian TCP-space, the target position is usually defined in RPY (roll, pitch, yaw) notation. However, KISMET allows to use other notations, e.g. EULER angles, Siemens-RCM notation etc.. Using TCP-space is a robot independant method of defining motions.

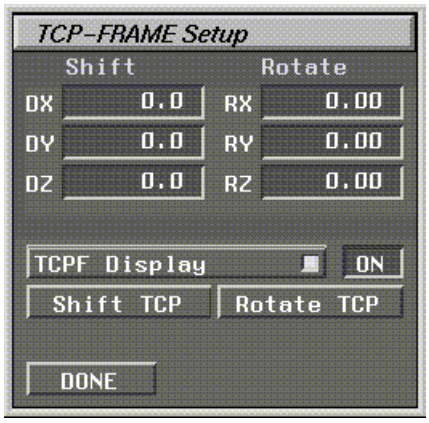

Figure 10: TCP-Frame Setup Panel

- **Shift TCP:**

  The TCP-frame is translated relative to its reference frame in 3 dimensions.

  - **LM** Translate TCP-frame about DX along reference frame X-axis (n)
  - **MM** Translate TCP-frame about DY along reference frame Y-axis (s)
  - **BM** Translate TCP-frame about DZ along reference frame Z-axis (a)

- **Rotate TCP:**

  The TCP-frame is rotated relative to its reference frame in 3 dimensions.

  - **LM** Rotate TCP-frame around reference frame X-axis (n)
  - **MM** Rotate TCP-frame around reference frame Y-axis (s)
  - **BM** Rotate TCP-frame around reference frame Z-axis (a)

7.2.2 The ZPF Setup Panel

This submenu is used for definition of the active robots ZEROPOINT (ZP) coordinate system offset parameters. The ZP-offset shifts the robot reference system (not the body) relative to the default position, which is perpendicular to the robot base coordinate system. Some robot controllers use the ZP-offset to make programming more efficient. For example, you can execute a robot subroutine with different ZP-frame positions. A typical application for this is palletising.

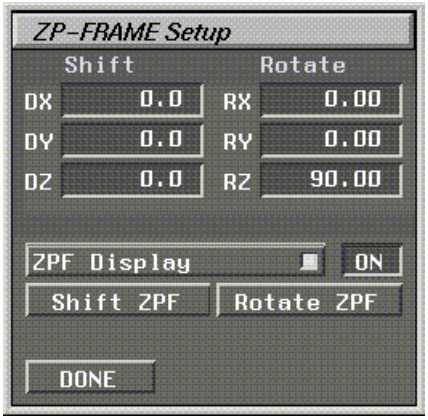

Figure 11: The ZP-Frame Setup Panel

- **Shift ZPF:**

  The ZeroPoint-frame is translated relative to its reference frame in 3 dimensions.

  - **LM** Translate ZP-frame about DX along reference frame X-axis
  - **MM** Translate ZP-frame about DY along reference frame Y-axis
  - **BM** Translate ZP-frame about DZ along reference frame Z-axis

- **Rotate ZPF:**

  The ZeroPoint-frame is rotated relative to its reference frame in 3 dimensions.

  - **LM** Rotate ZP-frame around reference frame X-axis
  - **MM** Rotate ZP-frame around reference frame Y-axis
  - **BM** Rotate ZP-frame around reference frame Z-axis