

mm\_trj  
V.1.0.1

Generated by Doxygen 1.8.5

Sun Aug 25 2013 16:59:23



# Contents

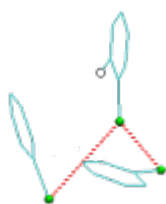
<b>1</b>	<b>mm_trj</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.2	How to use . . . . .	1
<b>2</b>	<b>Install</b>	<b>3</b>
2.1	Requirements . . . . .	3
2.2	How to install . . . . .	3
2.2.1	Linux . . . . .	3
2.2.2	Windows . . . . .	3
<b>3</b>	<b>Changelog</b>	<b>5</b>
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>File Documentation</b>	<b>9</b>
5.1	src/add_main.c File Reference . . . . .	9
5.1.1	Function Documentation . . . . .	9
5.1.1.1	error_checking . . . . .	9
5.1.1.2	print_message . . . . .	10
5.1.1.3	set_defaults . . . . .	10
5.2	src/atom_types.c File Reference . . . . .	10
5.2.1	Function Documentation . . . . .	11
5.2.1.1	reading_atoms . . . . .	11
5.3	src/main.c File Reference . . . . .	11
5.3.1	Function Documentation . . . . .	11
5.3.1.1	main . . . . .	11
5.4	src/messages.c File Reference . . . . .	12
5.4.1	Function Documentation . . . . .	12
5.4.1.1	message . . . . .	12
5.5	src/print_trj.c File Reference . . . . .	12
5.5.1	Function Documentation . . . . .	13
5.5.1.1	printing_trj . . . . .	13

---

5.6	src/read_gmx.c File Reference . . . . .	13
5.6.1	Function Documentation . . . . .	13
5.6.1.1	rw_gmx . . . . .	13
5.6.1.2	translate_coords . . . . .	14
5.7	src/read_puma.c File Reference . . . . .	14
5.7.1	Function Documentation . . . . .	14
5.7.1.1	rw_puma . . . . .	14
 <b>Index</b>		 <b>16</b>

# Chapter 1

## mm\_trj



### 1.1 Introduction

#### About this program:

- Program that generates trajectory files

#### Developer:

- Evgeniy Alekseev aka arcanis

<esalexeev (at) gmail (dot) com>

#### License:

- GPL

### 1.2 How to use

#### Usage:

```
mm_trj -i INPUT_TRJ -t INPUT_TYPE -s NUMBER -a INPUT_ATOMS -o OUTPUT [ -tt TOTAL_TYPES ]  
                                           [ -l LOGFILE ] [ -q ] [ -h ]
```

#### Parameters:

-i	- input file name
-t	- type of trajectory. Supported formats: gmx, puma
-s	- number of trajectory steps (integer)
-a	- input file with atom types. See file format in manual
-o	- mask of output files
-tt	- number of different atom types. Default is 1024
-l	- log enable
-q	- quiet enable
-h	- show this help and exit



# Chapter 2

## Install

### 2.1 Requirements

The application mm\_trj requires the following external stuff:

- cmake  $\geq$  2.8
- gcc  $\geq$  4.8

### 2.2 How to install

#### 2.2.1 Linux

```
* mkdir build && cd build
* cmake -DCMAKE_INSTALL_PREFIX=/usr -DCMAKE_BUILD_TYPE=Release ../
* make
* make install
*
```

#### 2.2.2 Windows

```
* create project file using 'cmake'
* compile project
*
```

You may also download compiled executable file for Win\_x86.





## Chapter 3

# Changelog

V.1.0.1 (2013-07-27)

- initial release



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

src/ <a href="#">add_main.c</a> . . . . .	9
src/ <a href="#">atom_types.c</a> . . . . .	10
src/ <a href="#">main.c</a> . . . . .	11
src/ <a href="#">messages.c</a> . . . . .	12
src/ <a href="#">print_trj.c</a> . . . . .	12
src/ <a href="#">read_gmx.c</a> . . . . .	13
src/ <a href="#">read_puma.c</a> . . . . .	14



## Chapter 5

# File Documentation

### 5.1 src/add\_main.c File Reference

```
#include <stdio.h>
#include "messages.h"
```

#### Functions

- int [error\\_checking](#) (const char \*input, const char \*input\_at, const char \*output, const int step, const int type)  
*function that checks errors in input variables*
- int [print\\_message](#) (const int quiet, FILE \*std\_output, const int log, FILE \*f\_log, const int mode, const char \*str)  
*function that prints message in log and stdout*
- int [set\\_defaults](#) (char \*input, char \*input\_at, int \*log, char \*output, int \*step, int \*total\_types, int \*type, int \*quiet)  
*function that sets default values of variables*

#### 5.1.1 Function Documentation

##### 5.1.1.1 int error\_checking ( const char \* input, const char \* input\_at, const char \* output, const int step, const int type )

function that checks errors in input variables

```
* error\_checking (cell, from, input, num_needed_at, needed_at, output, to);
*
```

#### Parameters

<i>input</i>	input file name
<i>input_at</i>	input file name with atom types
<i>output</i>	output file name
<i>step</i>	number of trajectory steps
<i>type</i>	type of trajectory

**Returns**

- 11 - error in 'input\_at'
- 12 - error in 'input'
- 13 - error in 'output'
- 14 - error in 'step'
- 15 - error in 'type'
- 0 - exit without errors

**5.1.1.2** `int print_message ( const int quiet, FILE * std_output, const int log, FILE * f_log, const int mode, const char * str )`

function that prints message in log and stdout

```
* print_message (quiet, stdout, log, f_log, 0, str);
*
```

**Parameters**

<i>quiet</i>	status of quiet-mode
<i>std_output</i>	stdout
<i>log</i>	status of log-mode
<i>f_log</i>	log file
<i>mode</i>	number of message in "messages.c"
<i>str</i>	additional text in message

**Returns**

- 0 - exit without errors

**5.1.1.3** `int set_defaults ( char * input, char * input_at, int * log, char * output, int * step, int * total_types, int * type, int * quiet )`

function that sets default values of variables

```
* set_defaults (input, input_at, &log, output, &step, &type, &quiet);
*
```

**Parameters**

<i>input</i>	input file name
<i>input_at</i>	input file name with atom types
<i>log</i>	status of log-mode
<i>output</i>	output file name
<i>step</i>	number of trajectory steps
<i>total_types</i>	number of different atom types
<i>type</i>	type of trajectory
<i>quiet</i>	status of quiet-mode

**Returns**

- 0 - exit without errors

## 5.2 src/atom\_types.c File Reference

```
#include <stdio.h>
```

## Functions

- int [reading\\_atoms](#) (const char \*input\_at, int \*num\_types, int \*num\_mol, int \*num\_atoms, char \*ch\_atom\_types, int \*atom\_types, const int total\_types)  
*function that reads atom types from input file*

### 5.2.1 Function Documentation

5.2.1.1 int [reading\\_atoms](#) ( const char \* *input\_at*, int \* *num\_types*, int \* *num\_mol*, int \* *num\_atoms*, char \* *ch\_atom\_types*, int \* *atom\_types*, const int *total\_types* )

function that reads atom types from input file

```
* reading\_atoms (input_at, &num_types, num_mol, num_atoms, ch_atom_types, atom_types,
*                 total_types);
*
```

#### Parameters

<i>input_at</i>	input file name with atom types
<i>num_types</i>	number of molecule types
<i>num_mol</i>	massive of number of molecules of selected type
<i>num_atoms</i>	massive of number of atoms of selected molecule
<i>ch_atom_types</i>	massive of char atom types
<i>atom_types</i>	massive of atom types
<i>total_types</i>	number of different atom types

#### Returns

- 1 - error in opening file
- 2 - error in file format
- 3 - memory error
- 0 - exit without errors

## 5.3 src/main.c File Reference

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "add_main.h"
#include "atom_types.h"
#include "messages.h"
#include "print_trj.h"
#include "read_gmx.h"
#include "read_puma.h"
```

## Functions

- int [main](#) (int argc, char \*argv[])

### 5.3.1 Function Documentation

5.3.1.1 int [main](#) ( int *argc*, char \* *argv*[] )

**Returns**

- 1 - error in error\_checking
- 2 - input file does not exist
- 3 - memory error
- 4 - unknown flag
- 0 - exit without errors

**5.4 src/messages.c File Reference**

```
#include <stdio.h>
#include <time.h>
```

**Functions**

- int [message](#) (const int log, const int mode, const char \*text, FILE \*output)  
*function that prints messages to output*

**5.4.1 Function Documentation**

**5.4.1.1** int [message](#) ( const int *log*, const int *mode*, const char \* *text*, FILE \* *output* )

function that prints messages to output

```
* message (log, mode, text, output);
*
```

**Parameters**

<i>log</i>	equal to 1 if print to logfile
<i>mode</i>	number of message
<i>text</i>	additional text
<i>output</i>	output file (may be stdout)

**Returns**

- 1 - unknown mode
- 0 - exit without errors

**5.5 src/print\_trj.c File Reference**

```
#include <stdio.h>
```

**Functions**

- int [printing\\_trj](#) (const char \*filename, const int atoms, const int num\_types, const int \*num\_mol, const int \*num\_atoms, const char \*ch\_atom\_types, const int \*atom\_types, const float \*coords)  
*function that prints trajectory snapshots*



### 5.5.1 Function Documentation

5.5.1.1 `int printing_trj ( const char * filename, const int atoms, const int num_types, const int * num_mol, const int * num_atoms, const char * ch_atom_types, const int * atom_types, const float * coords )`

function that prints trajectory snapshots

```
* printing_trj (filename, atoms, num_types, num_mol, num_atoms, ch_atom_types,
*               atom_types, coords);
*
```

#### Parameters

<i>filename</i>	output file name
<i>atoms</i>	number of atoms in system
<i>num_types</i>	number of molecule types
<i>num_mol</i>	massive of number of molecule of selected type
<i>num_atoms</i>	massive of number of atoms of selected molecule
<i>ch_atom_types</i>	massive of char atom types
<i>atom_types</i>	massive of atom types
<i>coords</i>	massive of coordinates

#### Returns

0 - exit without errors

## 5.6 src/read\_gmx.c File Reference

```
#include <math.h>
#include <stdio.h>
#include "print_trj.h"
```

### Functions

- `int translate_coords (const float coords, const float cell, float *trans)`  
*function that translates coordinate*
- `int rw_gmx (const char *input, const int step, const char *output, const int num_types, const int *num_mol, const int *num_atoms, const char *ch_atom_types, const int *atom_types, float *coords)`  
*function that read GROMACS trajectory file and write to output*

### 5.6.1 Function Documentation

5.6.1.1 `int rw_gmx ( const char * input, const int step, const char * output, const int num_types, const int * num_mol, const int * num_atoms, const char * ch_atom_types, const int * atom_types, float * coords )`

function that read GROMACS trajectory file and write to output

```
* rw_gmx (input, step, output, num_types, num_mol, num_atoms, ch_atom_types,
*         atom_types, coords);
*
```

## Parameters

<i>input</i>	input file name
<i>step</i>	number of trajectory steps
<i>output</i>	mask of output files
<i>num_types</i>	number of molecule types
<i>num_mol</i>	massive of number of molecule of selected type
<i>num_atoms</i>	massive of number of atoms of selected molecule
<i>ch_atom_types</i>	massive of char atom types
<i>atom_types</i>	massive of atom types
<i>coords</i>	massive of coordinates

## Returns

- 1 - file does not exist
- 0 - exit without errors

5.6.1.2 int translate\_coords ( const float *coords*, const float *cell*, float \* *trans* )

function that translates coordinate

```
* translate_coords (coords[3*i+j], cell[j], trans);
*
```

## Parameters

<i>coords</i>	coordinate
<i>cell</i>	cell size
<i>trans</i>	massive of translated coordinates

## Returns

- 0 - exit without errors

## 5.7 src/read\_puma.c File Reference

```
#include <stdio.h>
#include "print_trj.h"
```

## Functions

- int [rw\\_puma](#) (const char \**input*, const int *step*, const char \**output*, const int *num\_types*, const int \**num\_mol*, const int \**num\_atoms*, const char \**ch\_atom\_types*, const int \**atom\_types*, float \**coords*)  
function that read PUMA trajectory file and write to output

## 5.7.1 Function Documentation

5.7.1.1 int rw\_puma ( const char \* *input*, const int *step*, const char \* *output*, const int *num\_types*, const int \* *num\_mol*, const int \* *num\_atoms*, const char \* *ch\_atom\_types*, const int \* *atom\_types*, float \* *coords* )

function that read PUMA trajectory file and write to output

```
* rw_puma (input, step, output, num_types, num_mol, num_atoms, ch_atom_types,
*          atom_types, coords);
*
```

## Parameters

<i>input</i>	input file name
<i>step</i>	number of trajectory steps
<i>output</i>	mask of output files
<i>num_types</i>	number of molecule types
<i>num_mol</i>	massive of number of molecule of selected type
<i>num_atoms</i>	massive of number of atoms of selected molecule
<i>ch_atom_types</i>	massive of char atom types
<i>atom_types</i>	massive of atom types
<i>coords</i>	massive of coordinates

## Returns

- 1 - file does not exist
- 0 - exit without errors

# Index

- add\_main.c
  - error\_checking, [9](#)
  - print\_message, [10](#)
  - set\_defaults, [10](#)
- atom\_types.c
  - reading\_atoms, [11](#)
- error\_checking
  - add\_main.c, [9](#)
- main
  - main.c, [11](#)
- main.c
  - main, [11](#)
- message
  - messages.c, [12](#)
- messages.c
  - message, [12](#)
- print\_message
  - add\_main.c, [10](#)
- print\_trj.c
  - printing\_trj, [13](#)
- printing\_trj
  - print\_trj.c, [13](#)
- read\_gmx.c
  - rw\_gmx, [13](#)
  - translate\_coords, [14](#)
- read\_puma.c
  - rw\_puma, [14](#)
- reading\_atoms
  - atom\_types.c, [11](#)
- rw\_gmx
  - read\_gmx.c, [13](#)
- rw\_puma
  - read\_puma.c, [14](#)
- set\_defaults
  - add\_main.c, [10](#)
- src/add\_main.c, [9](#)
- src/atom\_types.c, [10](#)
- src/main.c, [11](#)
- src/messages.c, [12](#)
- src/print\_trj.c, [12](#)
- src/read\_gmx.c, [13](#)
- src/read\_puma.c, [14](#)
- translate\_coords
  - read\_gmx.c, [14](#)