rosbag/ Commandline

The rosbag command-line tool provides functionality for ROS bags.

It can record a bag, republish the messages from one or more bags, summarize the contents of a bag, check a bag's message definitions, filter a bag's messages based on a Python expression, compress and decompress a bag and rebuild a bag's index.

This is the current list of supported commands:

record
: Record a bag file with the contents of specified topics.

info
: Summarize the contents of a bag file.

play
: Play back the contents of one or more bag files.

check
: Determine whether a bag is playable in the current system, or if it can be migrated.

fix
: Repair the messages in a bag file so that it can be played in the current system.

filter
: Convert a bag file using Python expressions.

compress
: Compress one or more bag files.

decompress
: Decompress one or more bag files.

reindex
: Reindex one or more broken bag files.

These are described in greater detail in the following sections.

1. rosbag record

rosbag record subscribes to topics and writes a bag file with the contents of all messages published on those topics. The file contains interlaced, serialized ROS messages dumped directly to a single file as they come in over the wire. This is the most performance and disk-friendly recording format possible. To further reduce disk usage, you can compress bag files as they are created.

If you are recording messages at a high bandwidth, such as from cameras, it is strongly recommended you run rosbag record on the same machine as the camera, and specify the file destination as being on the local machine disk.

record <topic-names>
: Record a bag file with the contents of the specified topics.

```
$ rosbag record rosout tf cmd_vel
```

-h, --help
: Show the usage and exit.

```
$ rosbag record -h
```

-a, --all
: Record all topics.

```
$ rosbag record -a
```

Note that newly published topics are discovered by periodically polling the master. rosbag record -a will likely

miss initial messages published on any topic.

-e, --regex

Match topics using regular expressions.

```
$ rosbag record -e "/(.*)_stereo/(left|right)/image_rect_color"
```

Record all topics that contain sensors in their name:

```
$ rosbag record -e "(.*)sensors(.*)"
```

-p, --publish **New in ROS Melodic**

Publish a message when the record begin to write a new bag (topic = "begin_write").

```
$ rosbag record -p
```

-x EXCLUDE_REGEX, --exclude=EXCLUDE=REGEX

Exclude topics matching the given regular expression (subtracts from -a or -e).

```
$ rosbag record -e "/wide_stereo(.*)" -x "(.*)/points(.*)"
```

-q, --quiet

Suppress console output.

```
$ rosbag record -q /chatter
```

-d, --duration

Specify the maximum duration of the recorded bag file.

```
$ rosbag record --duration=30 /chatter
$ rosbag record --duration=5m /chatter
$ rosbag record --duration=2h /chatter
```

-o PREFIX, --output-prefix=PREFIX

Prepend PREFIX to beginning of bag name before date stamp.

```
$ rosbag record -o session1 /chatter
```

-O NAME, --output-name=NAME

Record to bag with name NAME.bag.

```
$ rosbag record -O session2_090210.bag /chatter
```

--split

Split the bag when maximum size or duration is reached

```
$ rosbag record --split --size=1024 /chatter
$ rosbag record --split --duration=30 /chatter
$ rosbag record --split --duration=5m /chatter
$ rosbag record --split --duration=2h /chatter
```

--max-splits=MAX_SPLITS **New in ROS Kinetic**

Split bag at most MAX_SPLITS times, then begin deleting the oldest files. This creates a fixed size or duration recording.

```
$ rosbag record --split --size 1024 --max-splits 3 /chatter
$ rosbag record --split --duration 10m --max-splits 6 /chatter
```

-b SIZE, --buffsize=SIZE

Use an internal buffer of SIZE MB (Default: 256, 0 = infinite). This is the message queue of the recorder object, before messages are being passed on to the bag. Lowering this value might result in messages being dropped before they reach the recording process.

```
$ rosbag record -b 1024 /chatter
```

--chunksize=SIZE

Advanced. Record to chunks of SIZE KB (Default: 768). This is a buffer within the bag file object. Lowering this value will result in more writes to disk.

```
$ rosbag record --chunksize=1024 /chatter
```

-l NUM, --limit=NUM

Only record NUM messages on each topic.

```
$ rosbag record -l 1000 /chatter
```

--node=NODE

Record all topics subscribed to by a specific node

```
$ rosbag record --node=/joy_teleop
```

-j, --bz2

Use BZ2 compression. See compress for details.

```
$ rosbag record -j /chatter
```

--lz4

Use LZ4 compression. See compress for details.

```
$ rosbag record --lz4 /chatter
```

2. rosbag info

rosbag info displays a human-readable summary of the contents of the bag files, including start and end times, topics with their types, message counts and median frequency, and compression statistics.

To output a machine-readable representation, use --yaml. See the cookbook for an example of how to load this representation in code.

info <bag-files>

Display a summary of the contents of the bag files.

```
$ rosbag info session*.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag info -h
```

-y, --yaml

Print information in YAML format.

```
$ rosbag info -y /path/to/my.bag
```

-k KEY, --key=KEY

Print information only on the given field (requires -y).

```
$ rosbag info -y -k duration /path/to/my.bag
```

Example usage:

```
$ rosbag info foo.bag
path:        foo.bag
version:     2.0
duration:    1.2s
start:       Jun 17 2010 14:24:58.83 (1276809898.83)
end:         Jun 17 2010 14:25:00.01 (1276809900.01)
size:        14.2 KB
messages:    119
compression: none [1/1 chunks]
types:       geometry_msgs/Point [4a842b65f413084dc2b10fb484ea7f17]
topics:      /points   119 msgs @ 100.0 Hz : geometry_msgs/Point
```

3. rosbag play

rosbag play reads the contents of one or more bag file, and plays them back in a time-synchronized fashion. Time synchronization occurs based on the global timestamps at which messages were received. Playing will begin immediately, and then future messages will be published according to the relative offset times. If two separate bag files are used, they are treated as a single bag with interlaced times according to the timestamps. This means if you record one bag, wait an hour, and record a second bag, when you play them back together you will have an hour-long dead period in the middle of your playback.

If you do not want to observe playback timing, the -i option will playback all messages from the file as fast as possible. Note that for large files this will often lead to exceeding your incoming buffers.

Additionally, during playing, you can pause at any time by pressing space. When paused, you can step through messages by pressing s.

play <bag-files>

Play back (publish) the contents of the given bags.

```
$ rosbag play recorded1.bag recorded2.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag play -h
```

-q, --quiet

Suppress console output.

```
$ rosbag play -q recorded1.bag
```

-i, --immediate

Play back all messages without waiting.

```
$ rosbag play -i recorded1.bag
```

--pause

Start in paused mode.

```
$ rosbag play --pause recorded1.bag
```

--queue=SIZE

Use an outgoing queue of size SIZE (defaults to 0. As of **1.3.3** defaults to 100).

```
$ rosbag play --queue=1000 recorded1.bag
```

--clock

Publish the clock time

```
$ rosbag play --clock recorded1.bag
```

--hz=HZ

Publish clock time at frequency HZ Hz (default: 100).

```
$ rosbag play --clock --hz=200 recorded1.bag
```

-d SEC, --delay=SEC

Sleep SEC seconds after every advertise call (to allow subscribers to connect).

```
$ rosbag play -d 5 recorded1.bag
```

-r FACTOR, --rate=FACTOR

Multiply the publish rate by FACTOR.

```
$ rosbag play -r 10 recorded1.bag
```

-s SEC, --start=SEC

Start SEC seconds into the bags.

```
$ rosbag play -s 5 recorded1.bag
```

-u SEC, --duration=SEC

Play only SEC seconds from the bag files.

```
$ rosbag play -u 240 recorded1.bag
```

--skip-empty=SEC

Skip regions in the bag with no messages for more than SEC seconds.

```
$ rosbag play --skip-empty=1 recorded1.bag
```

-l, --loop

Loop playback.

```
$ rosbag play -l recorded1.bag
```

-k, --keep-alive

Keep alive past end of bag (useful for publishing latched topics).

```
$ rosbag play -k recorded1.bag
```

4. rosbag check

check <bag-file>

Determine whether or not a bag is playable in the current system.

```
$ rosbag check old.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag check -h
```

-g RULEFILE, --genrules=RULEFILE

Generate a bag migration rule file named RULEFILE.

```
$ rosbag check -g diagnostics.bmr diag.bag
```

-a, --append

Append to the end of an existing bag migration rule file after loading.

```
$ rosbag check -a -g all.bmr diag.bag
```

-n, --noplugins

Do not load rule files via plugins.

```
$ rosbag check -n diag.bag
```

If a bag is currently unplayable and non-migratable, it may be necessary to generate rules to bring the bag up to date. See Migrating Messages for the preferred approaches to doing this.

5. rosbag fix

fix <in-bag> <out-bag> [rules.bmr]

Repairs a bag using registered rules (and optionally locally defined rules).

```
$ rosbag fix old.bag repaired.bag myrules.bmr
```

-h, --help

Show the usage and exit.

```
$ rosbag fix -h
```

-n, --noplugins

Do not load rule files via plugins.

```
$ rosbag fix -n old.bag repaired.bag
```

6. rosbag filter

filter <in-bag> <out-bag> <expression>

Convert a bag file using the given Python expression.

```
$ rosbag filter my.bag only-tf.bag "topic == '/tf'"
```

The Python expression can access any of the Python builtins plus:

topic

the topic of the message

m

the message

t

time of message. The time is represented as a rospy Time object (t.secs, t.nsecs)

-h, --help

Show the usage and exit.

```
$ rosbag filter -h
```

--print=PRINT-EXPRESSION

Evaluate and print a Python expression for verbose debugging. Uses same variables as filter expression.

```
$ rosbag filter --print='"%s @ %d.%d: %s' % (topic, t.secs, t.nsecs, m.data)" big.bag small.bag "topic == '/chatter'"
```

Example output:

```
NO MATCH hello world 68
NO MATCH hello world 69
NO MATCH hello world 70
MATCH hello world 71
NO MATCH hello world 72
NO MATCH hello world 73
```

To filter based on time, convert the time to a floating point number (use UNIX time, to get this value, use rosbag info):

```
rosbag filter input.bag output.bag "t.to_sec() <= 1284703931.86"
```

7. rosbag compress

rosbag compress is a command-line tool for compressing bag files.

A backup of each bag file (with the extension .orig.bag) is made before the bag is compressed. If the backup file already exists (and the -f option isn't specified) then the tool will not compress the file.

Currently, there are two supported formats: BZ2 and LZ4. BZ2 is selected by default. Which should you prefer? BZ2 generally produces smaller bags than LZ4, so use it if disk usage is your primary concern. However, BZ2 is typically much slower than LZ4, both during compression and later when a bag's chunks are decompressed for reading, so use it if you're willing to tolerate slightly larger bags. Look here a more detailed discussion and some benchmarks.

The bag format supports interleaving compressed data chunks with uncompressed chunks in a single bag. Running rosbag compress on a bag with compressed data chunks will decompress and compress the bag using the specified compression format.

compress <bag-files>

Compress the given bag files using BZ2.

```
$ rosbag compress *.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag compress -h
```

--output-dir=DIR

Write to directory DIR.

```
$ rosbag compress --output-dir=compressed *.bag
```

-f, --force

Force overwriting of backup file it it exists.

```
$ rosbag compress -f *.bag
```

-q, --quiet

Suppress noncritical messages.

```
$ rosbag compress -q *.bag
```

-j, --bz2

Use BZ2 to compress data.

```
$ rosbag compress -j *.bag
```

--lz4

Use LZ4 to compress data.

```
$ rosbag compress --lz4 *.bag
```

8. rosbag decompress

rosbag decompress is a command-line tool for decompressing bag files. It automatically determines which compression format a bag uses.

A backup of each bag file (with the extension .orig.bag) is made before the bag is decompressed. If the backup file already exists (and the -f option isn't specified) then the tool will not decompress the file.

decompress <bag-files>

Decompress the given bag files.

```
$ rosbag decompress *.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag decompress -h
```

--output-dir=DIR

Write to directory DIR.

```
$ rosbag decompress --output-dir=uncompressed *.bag
```

-f, --force

Force overwriting of backup file it it exists.

```
$ rosbag decompress -f *.bag
```

-q, --quiet

Suppress noncritical messages.

```
$ rosbag decompress -q *.bag
```

9. rosbag reindex

rosbag reindex is a command-line tool for repairing broken bag files (or bag files recorded prior to ROS version 0.11.) If a bag was not closed cleanly for any reason, then the index information may be corrupted. Use this tool to reread the message data and rebuild the index.

A backup of each bag file (with the extension .orig.bag) is made before the bag is reindexed. If the backup file already exists (and the -f option isn't specified) then the tool will not reindex the file.

reindex <bag-files>

Reindex the given bag files.

```
$ rosbag reindex *.bag
```

-h, --help

Show the usage and exit.

```
$ rosbag reindex -h
```

--output-dir=DIR

Write to directory DIR.

```
$ rosbag reindex --output-dir=reindexed *.bag
```

-f, --force

Force overwriting of backup file it it exists.

```
$ rosbag reindex -f *.bag
```

-q, --quiet

Suppress noncritical messages.

```
$ rosbag reindex -q *.bag
```