

Generate Debian packages for the ROS packages in Autoware.core and Autoware.universe #3222

New issue

Open

29 of 31 tasks

esteve opened this issue on Jan 24, 2023 · 81 comments



esteve commented on Jan 24, 2023 ·

edited

Contributor

Checklist

- ☒ I've read the [contribution guidelines](#).
- ☒ I've searched other issues and no duplicate issues were found.
- ☒ I've agreed with the maintainers that I can plan this task.

Description

Given that Ubuntu is the platform of choice for developing and running Autoware, providing binary Debian packages will help users deploying Autoware

Purpose

Provide Debian packages for our users and developers

Possible approaches

Integrate <https://github.com/jspricke/ros-deb-builder-action> into our GitHub actions workflow

Definition of done

Debian packages are generated for all ROS packages and can be downloaded from our repositories

- ☒ [build\(tamagawa_iwu_driver\): added missing dependency tier4/tamagawa_imu_driver#9](#)
- ☒ [build\(ndt_omp\): add ros_environment dependency tier4/ndt_omp#20](#)
- ☒ [fix\(autoware_auto_planning_msgs\): added action_msgs](#)

Assignees

- esteve
- isamu-takagi

Labels

type:installation type:new-feature

Projects

None yet

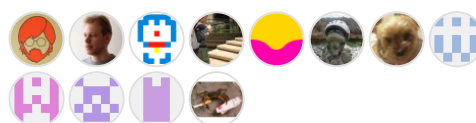
Milestone

No milestone

Development

No branches or pull requests

12 participants



- [dependency tier4/autoware_auto_msgs#37](#)
- ✓ [🔗 build\(system_monitor\): add build dependency autoware.universe#2740](#)
- ✓ [🔗 build\(dummy_infrastructure\): add build dependency autoware.universe#2739](#)
- ✓ [🔗 build\(bluetooth_monitor\): add build dependency autoware.universe#2738](#)
- ✓ [🔗 build\(gnss_poser\): add build dependency autoware.universe#2737](#)
- ✓ [🔗 build\(signal_processing\): add build dependency autoware.universe#2736](#)
- ✓ [🔗 build\(detected_object_validation\): add build dependency autoware.universe#2735](#)
- ✓ [🔗 build\(kalman_filter\): add build dependency autoware.universe#2734](#)
- ✓ [🔗 build\(autoware_auto_perception_rviz_plugin\): add build dependency autoware.universe#2733](#)
- ✓ [🔗 build\(freespace_planning_algorithms\): add nlohmann-json-dev dependency autoware.universe#2818](#)
- ✓ [🔗 Added apt-cacher-ng and auto-apt-proxy to cache downloaded Debian packages jspricke/ros-deb-builder-action#9](#)
- ✓ [🔗 Do not rebuild packages that have been already generated if the -c option is passed jspricke/ros-deb-builder-action#10](#)
- ✓ [🔗 Added python3-open3d via .deb package ros/rosdistro#36052](#)
- ✓ [🔗 build\(lidar_centerpoint\): use APT version of python3-open3d as dependency autoware.universe#2869](#)
- ✓ [🔗 build\(system_monitor\): added missing Boost dependencies autoware.universe#2881](#)
- ✓ [🔗 build\(signal_processing\): add Boost to build_export_depend autoware.universe#2917](#)
- ✓ ~~[🔗 build\(obstacle_avoidance_planner\): added libopencv-dev dependency autoware.universe#2936](#)~~
- ✓ [🔗 build\(obstacle_stop_planner\): added libopencv-dev dependency autoware.universe#2937](#)
- ✓ [🔗 build\(pacmod_interface\): added missing dependencies tier4/pacmod_interface#54](#)
- ✓ ~~[🔗 fix: add pacmod3_msgs repository for pacmod_interface #3287](#)~~
- ✓ [🔗 build\(trjectory_follower_node\): added missing dependencies autoware.universe#3026](#)
- ✓ [🔗 build\(autoware_testing\): add missing dependencies autoware.universe#3093](#)
- ✓ [🔗 build\(bytetrack\): add missing dependencies autoware.universe#3280](#)

Related issues:

- ☒ [Move downloading artifacts outside CMake](#) autoware.universe#3137
- ☐ [Release Autoware packages into the ROS buildfarm](#) #3360
- ☐ [Move CUDA-dependent packages to a separate repository](#) autoware.universe#3135

 3



esteve self-assigned this on Jan 24, 2023



esteve mentioned this issue on Jan 24, 2023

build: add missing build dependency

 Closed

autowarefoundation/autoware.universe
#2721

 4 tasks



isamu-takagi commented on Jan 24, 2023 •
edited ▼

Contributor

I made some changes to actions to support a differential build. I would like to decide a release operation of Autoware and propose features to upstream to support it.

<https://github.com/tier4/autoware-apt-testing/tree/main>
<https://github.com/isamu-takagi/ros-deb-builder-action/tree/develop> (contains a lot of temporary code)

The features I think we need are:

- support differential build
- support multiple suite/components/architectures
- use version in package.xml
- keep the old versions

So first, we need to decide how to release Autoware and how to manage the package version.

 1



isamu-takagi commented on Jan 24, 2023

Contributor

My idea is that each package will only build when the version tag in package.xml is changed, and the repository keeps all previously built versions.



esteve commented on Jan 24, 2023

Contributor

Author

@isamu-takagi thanks for listing all the features, really nice.

I prefer to follow what Movelt is doing:

https://github.com/moveit/moveit2_packages/blob/main/.github/workflows/build.yaml#L6

it runs once every day (in Movelt's case, at 1:17am), so there's no need for differential builds, because it doesn't runs when a new branch is merged. But I think using the version from package.xml would be a useful alternative to using tags, though it's strongly encouraged for ROS packages/repositories to use tags for every new release.



esteve commented on Jan 24, 2023

Contributor

Author

@isamu-takagi I tried adding you as an assignee to this issue, but for some reason your username does not show up on the list. @kenji-miyake do you know why I can't add @isamu-takagi ?



isamu-takagi commented on Jan 24, 2023 •

edited ▼

Contributor

it runs once every day (in Movelt's case, at 1:17am), so there's no need for differential builds, because it doesn't runs when a new branch is merged.

My concern is that the build takes a long time. If the dependent packages have not changed, it is efficient to use the released ones.

But I think using the version from package.xml would be a useful alternative to using tags, though it's strongly encouraged for ROS packages/repositories to use tags for every new release.

I think there is a request to use the old package version. in this case, If the package has not changed, the same binary will be generated as a different version.




kenji-miyake commented on Jan 24, 2023

Contributor

@kenji-miyake do you know why I can't add @isamu-takagi ?

@esteve Hmm, sorry I'm not sure. I believe you have
Maintain access to this repository.
For now, I'll add him.



 **kenji-miyake** assigned **isamu-takagi** on Jan 24, 2023



This was referenced on Jan 24, 2023

build(autoware_auto_perception_rviz_plugin): add build dependency
autowarefoundation/autoware.univers
e#2733

 Merged

build(kalman_filter): add build dependency
autowarefoundation/autoware.univers
e#2734

 Merged

build(detected_object_validation): add build dependency
autowarefoundation/autoware.univers
e#2735

 Merged

build(signal_processing): add build dependency
autowarefoundation/autoware.univers
e#2736

 Merged

build(gnss_poser): add build dependency
autowarefoundation/autoware.univers
e#2737

 Merged

build(bluetooth_monitor): add build dependency
autowarefoundation/autoware.univers
e#2738

 Merged

build(dummy_infrastructure): add build dependency
autowarefoundation/autoware.univers
e#2739

 Merged

build(system_monitor): add build dependency
autowarefoundation/autoware.univers
e#2740

 Merged



jspricke commented on Jan 24, 2023

My concern is that the build takes a long time. If the dependent packages have not changed, it is efficient to use the released ones.

I measured about a minute for a package that is already in ccache on the free Github VMs. We can speed it up by using a ram disk if we have more powerful VMs. Also we think about parallelizing the build.

Note that it is not enough to compare the diff in the repo itself as upstream dependencies could have changed and require a rebuild (ABI changes, message changes..).

But I think using the version from package.xml would be a useful alternative to using tags, though it's strongly encouraged for ROS packages/repositories to use tags for every new release.

I think there is a request to use the old package version. In this case, If the package has not changed, the same binary will be generated as a different version.

Currently the package includes the date as a Debian revision to make sure new versions are recognized as an update by apt. This is the same for the OSRF buildfarm. Generating a Debian revision only in case of a change is not trivial but we can discuss that.

Can you please send relevant patches upstream so we can discuss them?



esteve commented on Jan 25, 2023

Contributor

Author

Can you please send relevant patches upstream so we can discuss them?

We will when we feel like we have something that can be upstreamed, until then we'll continue with our experiments here.



isamu-takagi commented on Jan 25, 2023 •
edited ▼

Contributor

Can you please send relevant patches upstream so we can discuss them?

[isamu-takagi/ros-deb-builder-action#1](#)

This is the current diff, but not enough discussion. I would like to continue the discussion here.

Currently the package includes the date as a Debian revision to make sure new version are recognized as an update by apt. This is the same for the OSRF buildfarm. Generating a Debian revision only in case of a change is not trivial but we can discuss that.

The autoware.universe repository contains a lot of packages, so versioning with tags is not suitable. Multiple versions of deb packages for each tag is generated even if the code of stable package hasn't changed. So in current repository structure, we need versioning for each package.

If versioning by tag, the autoware.universe repository should be divided into multiple small repositories. This allows the repository/package maintainers to set the version at their release timing.

~~If using this method, It's need to deploy deb packages to another repository for distribution.~~

I noticed `sources.repos` resolve this.



jspricke commented on Jan 25, 2023

[isamu-takagi/ros-deb-builder-action#1](#) This is the current diff, but not enough discussion. I would like to continue the discussion here.

Fine with me. I'm also open to a video call if you are interested.

The autoware.universe repository contains a lot of packages, so versioning with tags is not suitable. Multiple versions of deb packages for each tag is generated even if the code of stable package hasn't changed. So in current repository structure, we need versioning for each package.

If versioning by tag, the autoware.universe repository should be divided into multiple small repositories. This allows the repository/package maintainers to set the version at their release timing.

I think that is a good idea in general (though unrelated to packaging) as monorepos do not work well with git.

~~If using this method, It's need to deploy deb packages to another repository for distribution.~~ I noticed `sources.repos` resolve this.

Not sure I got you here but it is also possible to pass extra apt repositories to sbuild via `SBUILD_CONF`.



1



isamu-takagi commented on Jan 25, 2023 •

Contributor

edited ▼

If versioning by tag, the autoware.universe repository should be divided into multiple small repositories. This allows the repository/package maintainers to set the version at their release timing.

What do you think about this? [@mitsudome-r](#)
[@yukkysaito](#) [@kenji-miyake](#) [@xmfcx](#)



esteve commented on Jan 25, 2023

Contributor

Author

Fine with me. I'm also open to a video call if you are interested.

We can do that, [@isamu-takagi](#) is based in Japan and I'm in Spain, I'll send you an email.



yukkysaito commented on Jan 25, 2023 •
edited ▾

Contributor

What do you think about this? [@mitsudome-r](#)
[@yukkysaito](#) [@kenji-miyake](#) [@xmfcx](#)

[@isamu-takagi](#)

I am thinking now...

On the negative side, some packages have dependencies between packages, and it is a pain to manage PR across packages when modifying functionality.

Are you considering a repository split for each package? Or per component?

Or instead of splitting the repository and releasing them separately, how about deb-package and release each one at the time of semantic version release on autoware?



1



kenji-miyake commented on Jan 25, 2023

Contributor

If versioning by tag, the autoware.universe repository should be divided into multiple small repositories. This allows the repository/package maintainers to set the version at their release timing.

What do you think about this? [@mitsudome-r](#)
[@yukkysaito](#) [@kenji-miyake](#) [@xmfcx](#)

[@isamu-takagi](#) I think flexible ways are better ways. I mean, reading the version in `package.xml` is suitable at least for Autoware as of today. If someone want to use Git tags, then they can update the version in `package.xml` when they create a tag.



1



jspricke commented on Jan 25, 2023

@isamu-takagi I think flexible ways are better ways. I mean, reading the version in `package.xml` is suitable at least for Autoware as of today. If someone want to use Git tags, then they can update the version in `package.xml` when they create a tag.

Note that the `package.xml` version does not accommodate changes in git so new commits will have the same version number. That's why we use `git describe --tags`.



isamu-takagi commented on Jan 26, 2023

Contributor

Note that the `package.xml` version does not accommodate changes in git so new commits will have the same version number. That's why we use `git describe --tags`.

What is the difference between tag and `package.xml`? New commits will have the same version unless a new tag is created. I think it can be replaced by just updating the `package.xml` instead of creating the tag. Did I miss something?



1



jspricke commented on Jan 26, 2023

Note that the `package.xml` version does not accommodate changes in git so new commits will have the same version number. That's why we use `git describe --tags`.

What is the difference between tag and `package.xml`? New commits will have the same version unless a new tag is created. I think it can be replaced by just updating the `package.xml` instead of creating the tag. Did I miss something?

Using `git describe --tags` will generate a version number specific to the used commit. It has the form `<tag>--<number of commits on top>-g<sha1 of commit>`. This makes sure that new commits have a higher version number (due to the counter) and that one can identify the commit belonging to the package (by the sha1).



jspricke commented on Jan 26, 2023

Note that this is mostly relevant when creating packages for non tagged versions such as nightly builds. For tagged release builds `git describe --tags` will resolve to the tag so it should be the same as what is in the package.xml



isamu-takagi commented on Jan 26, 2023

Contributor

I understood. At least, if it has a build time suffix as debian version, either tag or package.xml is fine. But using `git describe --tags` will give a commit hash.

50 hidden items

[Load more...](#)



jspricke commented on Mar 20, 2023

Thanks, that seems to be the case. Your solution seems fine, but I'm trying with `$core_depends` instead as it's cleaner (see https://manpages.debian.org/bullseye/sbuild/sbuild.conf.5.en.html#CORE_DEPENDS)

Interesting, can you share a log when it works for you?



xmfcx commented on Mar 20, 2023

Contributor

CUDA / TVM dependent packages

It's much simpler than that, we can add extra rosdep repositories via `ROSDEP_SOURCE`, no need to fully clone the ROS index and keep it synced, rosdep allows you to add multiple sources.

Oh it's even better then.

In any case, the ROS buildfarm and the action are not the same software, they have different documentation, although some things apply to both, so I wouldn't use the ROS buildfarm documentation as reference for the action, and viceversa.

Yes, I understand, thanks.

I still think it's cleaner to move the CUDA-related packages to a separate repository so that we have a set of packages that we know can be built with the dependencies currently available in ROS and that don't have any license issues.

I have 3 reasons to not move them into another repository:

1. We already have too many repositories and it makes it hard to keep them in sync.
 2. Right now the CUDA dependent packages are all belong to perception modules. But in future we could have many more packages from sensing, planning, control and localization components too. So having them in a separate place just from the CUDA dependency, doesn't seem right.
- We could have the usual component based folders in the `autoware_cuda` repository too.
3. In future, some packages may support both CUDA and non-CUDA in their implementations. Depending on the CUDA installation, the `CMakeLists.txt` file could follow 2 build paths.
- We should discuss if these kind of packages should be allowed. And recommend separating them into 2 packages.

But if everyone agrees, I'm ok with having a separate repository containing CUDA dependent repositories.



ambroise-arm commented on Mar 20, 2023

Collaborator

• edited ▼

TVM dependent packages

A few notes.

The current state of TVM support in Autoware doesn't allow to run inference using CUDA backend (at least not in the CI-enabled way we currently do it). There is a blocker in the deployment of the runtime TVM library that needs to be compiled with CUDA support, but then it prevents using the CUDA-enabled runtime library in a non-CUDA environment. Which goes against the plan of having a single [tvm_vendor](#) package (see [this discussion](#) for more info).

The currently supported backends (for Autoware packages) are: CPU, Vulkan and OpenCL (but not ROCm). To my knowledge, we don't have any performance figures between different backends. Vulkan is nice in that it has a software-only driver with [llvmpipe](#). Although it hasn't been tested for compatibility with the Autoware packages, it could allow to have a release of the TVM packages with something that can run both on CPU-only platforms and on GPU-accelerated platforms. Vulkan is also nice because it is supported by multiple GPU vendors.

In the future, as part of OpenADKit, the plan would be to allow compiling several backends as part of the package, with a runtime selection of which to use, but it will take some time before I am able to complete this. cc @kasperornmeck



esteve commented on Mar 20, 2023 Contributor Author

Interesting, can you share a log when it works for you?

@jspricke Bummer, it didn't work. The packages in `$core_depends` are installed after setting up the APT repositories, so installing `ca-certificates` via `$core_depends` won't fix the issue with astuff's repo not being trusted, but your solution works, thanks! 😊

Nevertheless, would it make sense to add `ca-certificates` to the `chroot` in <https://github.com/jspricke/ros-deb-builder-action/blob/main/prepare#L41-L43> ? We already have a couple of non-base packages and `ca-certificates` is in fact a `Recommends` dependency for `apt`



jspricke commented on Mar 20, 2023

Nevertheless, would it make sense to add `ca-certificates` to the `chroot` in <https://github.com/jspricke/ros-deb-builder-action/blob/main/prepare#L41-L43> ? We already have a couple of non-base packages and `ca-certificates` is in fact a `Recommends` dependency for `apt`

I was debating that with myself already, can you open a PR for that so we can discuss?



esteve commented on Mar 20, 2023 Contributor Author

I was debating that with myself already, can you open a PR for that so we can discuss?

[jspricke/ros-deb-builder-action#21](#)



esteve commented on Mar 21, 2023 Contributor Author
• edited ▾

Updates

Unsigned pacmod3 repository

It's now possible to build `pacmod_interface` with the latest changes in `ros-deb-builder-action` and the following:

```
ROSDEP_SOURCE: yaml
https://s3.amazonaws.com/autonomoustuff-
repo/autonomoustuff-public-humble.yaml
SBUILD_CONFIG: $extra_repositories = ["deb
[trusted=yes]
https://s3.amazonaws.com/autonomoustuff-repo/
focal main"];
```



ONNX model files

In addition to the problems with perception packages depending on CUDA, one thing I forgot to mention is that the following packages download the ONNX models via CMake's `file(DOWNLOAD ...)` in their `CMakeLists.txt` files:

- `image_projection_based_fusion`
- `lidar_centerpoint`
- `tensorrt_yolo`
- `tensorrt_yolox`
- `traffic_light_classifier`
- `traffic_light_ssd_fine_detector`

And I presume the `sbuild` `chroot` is either not configured to access the internet or network access is disabled for security reasons, so the ONNX models can't be downloaded.

We discussed this issue a while ago and decided to leave the `CMakeLists.txt` files as they are, but I think it's worth revisiting and evaluate `git-lfs` again (or similar). With `git-lfs` we ensure that a checkout belongs to a specific commit id, with all its artifacts and that its build can be reproduced consistently (see <https://reproducible-builds.org/> for why it's important that builds are reproducible).



xmfcx commented on Mar 21, 2023

Contributor

Create issues:

- Discuss about submitting the autoware repositories to the ros build farm
- Move and rename the velodyne driver package
- Separate cuda based packages
- Add download artifacts for the onnx files
<https://gitlab.com/autowarefoundation/autoware.auto/A>

[utowareAuto/-/blob/15d2e91a2b3c8cd61b20b7e6e79da32919c2cbcd/src/tools/autoware_auto_cmake/autoware_auto_cmake.cmake#L33](#)



2



jspricke commented on Mar 21, 2023 • edited ▾

And I presume the `sbuild` `chroot` is either not configured to access the internet or network access is disabled for security reasons, so the ONNX models can't be downloaded.

`sbuild --chroot-mode=unshare` uses `unshare` to open a new user name space without network access. This is to make sure the build works reproducibly and to comply with the Debian policy. There is currently no option to disable that.



This was referenced on Mar 22, 2023

Move CUDA-dependent packages to a separate repository

Open

autowarefoundation/autoware.universe#3135

Move downloading artifacts outside CMake

Closed

autowarefoundation/autoware.universe#3137

Release Autoware packages into the ROS buildfarm #3360

Open



esteve commented on Mar 22, 2023 Contributor Author

@xmfcx

Create issues:

- Discuss about submitting the autoware repositories to the ros build farm
- Move and rename the velodyne driver package
- Separate cuda based packages
- Add download artifacts for the onnx files
[https://gitlab.com/autowarefoundation/autoware.a
uto/AutowareAuto/-/blob/15d2e91a2b3c8cd61b20
b7e6e79da32919c2cbcd/src/tools/autoware_auto
_cmake/cmake/autoware_auto_cmake.cmake#L33](https://gitlab.com/autowarefoundation/autoware.auto/AutowareAuto/-/blob/15d2e91a2b3c8cd61b20b7e6e79da32919c2cbcd/src/tools/autoware_auto_cmake/cmake/autoware_auto_cmake.cmake#L33)

I've created the following issues:

- [🕒 Release Autoware packages into the ROS buildfarm #3360](#)
- [🕒 Move CUDA-dependent packages to a separate repository autoware.universe#3135](#)
- [👍 Move downloading artifacts outside CMake autoware.universe#3137](#)

Unfortunately, the velodyne repository (https://github.com/tier4/velodyne_vls) does not have an issues tab, [@kenji-miyake](#) could you or any of the owners of that repository enable that? Thanks.



xmfcx commented on Mar 22, 2023 • edited ▾

Contributor

[@esteve](#) Thanks for creating all the issues 🙏

I think we can keep the https://github.com/tier4/velodyne_vls as is, and fork it under autowarefoundation with a different name. I will do it right now.

Edit: https://github.com/autowarefoundation/velodyne_awf is forked now. And created the issue for it:

- https://github.com/autowarefoundation/velodyne_awf/issues/1



esteve commented on Mar 22, 2023 • edited ▾

Contributor

Author

[@esteve](#) Thanks for creating all the issues 🙏

[@xmfcx](#) you're welcome 😊

I think we can keep the https://github.com/tier4/velodyne_vls as is, and fork it under autowarefoundation with a different name. I will do it right now.

Thanks, but I think it'd be better to transfer the repository instead of forking it, so that the current maintainers of the TierIV fork can continue working on it. If we fork the TierIV fork, I worry we'll end up with two separate repositories with different commits and again we'll have to reconcile the changes at some point.



1



xmfcx commented on Mar 22, 2023 •

Contributor

edited ▼

Thanks, but I think it'd be better to transfer the repository instead of forking it, so that the current maintainers of the TierIV fork can continue working on it. If we fork the TierIV fork, I worry we'll end up with two separate repositories with different commits and again we'll have to reconcile the changes at some point.

Plan is to archive that TIER IV repository. That way old code will keep working if anyone depends on it. But new development can only be done on the AWF version.

Also this will give us time to transition to the new version in the launch files of Autoware.

cc: [@mitsudome-r](#) [@kenji-miyake](#)



jspricke commented on Mar 22, 2023

Plan is to archive that TIER IV repository. That way old code will keep working if anyone depends on it. But new development can only be done on the AWF version.

Also this will give us time to transition to the new version in the launch files of Autoware.

Transferring the repo in the Github web interface will add redirects from the old to the new address so everything keeps working. It makes it also easier to discover and is generally preferred.



2



xmfcx commented on Mar 22, 2023

Contributor

Ok tomorrow we can move the [repo](#) with [@mitsudome-r](#) then.



jiangqii commented on May 9, 2023

Do you have any plans to migrate to Redhat



xmfcx commented on May 10, 2023 •

Contributor

edited ▼

Do you have any plans to migrate to Redhat

Hello [@jiangqii](#) , we don't have any plans for it just yet.

But you might try the [Docker Installation Method](#) of the Autoware.



jiangqii commented on May 10, 2023

Do you have any plans to migrate to Redhat

Hello [@jiangqii](#) , we don't have any plans for it just yet.

But you might try the [Docker Installation Method](#) of the Autoware.

Thank you for your reply. I will try it on non Ubuntu platforms.



This was referenced on May 12, 2023

**feat(autoware.repos): change
velodyne driver repository to AWF**
#3492

Closed

**feat: rename velodyne_driver and
velodyne_pointcloud package name
in launch**
autowarefoundation/sample_sensor_kit
_launch#58

Closed

**feat: rename veldyne_driver and
velodyne_pointcloud package name**
tier4/awsim_sensor_kit_launch#7

Open



esteve mentioned this issue on May 18, 2023

**feat(source-installation):
recommend installation via Ansible
over manual installation**
autowarefoundation/autoware-
documentation#369

Merged

4 tasks



xmfcx added the **type:installation** label
on Nov 16, 2023

xmfcx commented on Nov 16, 2023

Contributor



[@esteve](#) I've been going over all the issues in the Autoware repository.

This issue is massive 🤯 and thanks for all your efforts throughout this work.

Could you summarize what still needs to be done, what steps are left?

I would like to help if possible.



 **xmfcx** mentioned this issue on Nov 16, 2023

Set up AWS EC2 instance for generating Debian packages nightly

✓ Closed

#3417

📋 3 tasks



esteve commented on Nov 16, 2023

Contributor

Author

[@xmfcx](#) work on generating Debian packages using the GitHub action is happening at <https://github.com/autowarefoundation/autoware-deb-packages>, we have packages for all of Autoware, but I'm currently working on adding support for CUDA.

However once we have a 1.0 release it'd be useful for users and for us to submit the packages in the Autoware repositories to the official ROS buildfarm so they can be included in future ROS distros and to offload the work to the ROS buildfarm.



 **xmfcx** added the `type:new-feature` label on Nov 28, 2023