It is fully autonomous, non-GUI-based, self-calibrating, and compliant with the VXWORKS flight software system.

This work was done by Lukas Mandrake, Benjamin J. Bornstein, Stojan Madzunkov, and John A. Macaskill of Caltech for NASA’s JPL. For more information, contact iaoffice@jpl.nasa.gov.

The software used in this innovation is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-46956.

Adaption of the AMDIS Method to Flight Status on the VCAM Instrument

Software has been developed to function onboard the International Space Station (ISS) to help safeguard human health by detecting compounds of concern in the cabin atmosphere, both in identity and concentration. This software calibrates and processes a standard 2D dataset (mass spectrum versus time) output from a gas chromatogram/mass spectrometer by identifying temporal events, including the possibility for near simultaneous event overlap, reducing the mass spectra for each event and comparing to an arbitrary library of known compounds. The level of autonomy, adjustment of parameters for the VCAM devices’ specific data characteristics, and adaptive mass resolution to ease requirement of precision mass calibration are three unique features of this design. The estimation of concentration is also a significant addition to the standard AMDIS (NIST) implementation. Solution filtration based on elution time, and an arbitration algorithm for similar matches, provide the user with a more succinct, single-valued estimate in comparison to algorithms designed to merely augment expert hand analysis.

This work was done by Lukas Mandrake, Benjamin J. Bornstein, Saungwon Lee, and Brian D. Bue of Caltech for NASA’s Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

The software used in this innovation is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-46956.

Natural Language Interface for Safety Certification of Safety-Critical Software

Model-based design and automated code generation are being used increasingly at NASA. The trend is to move beyond simulation and prototyping to actual flight code, particularly in the guidance, navigation, and control domain. However, there are substantial obstacles to more widespread adoption of code generators in such safety-critical domains. Since code generators are typically not qualified, there is no guarantee that their output is correct, and consequently the generated code still needs to be fully tested and certified.

The AutoCert generator plug-in supports the certification of automatically generated code by formally verifying that the generated code is free of different safety violations, by constructing an independently verifiable certificate, and by explaining its analysis in a textual form suitable for code reviews. This enables missions to obtain assurance about the safety and reliability of the code without excessive manual effort. The key technical idea is to exploit the idiomatic nature of auto-generated code in order to automatically infer logical annotations that describe properties of the code. These allow the automatic formal verification of the safety properties without requiring access to the internals of the code generator. The approach is therefore independent of the particular generator used. The use of a combined generation/analysis tool can allow system engineers to concentrate on the modeling and design, rather than worrying about low-level software details. By providing tracing between code and verification artifacts, and customizable safety reports, the tool supports both certification and debugging. Although integrated with the code generator, AutoCert is functionally independent in the sense that it does not rely on the correctness of any generator components. The tool has two main benefits: (1) it helps catch bugs in autcoders, and (2) it helps with the certification process for the autogenerated code, thus mitigating the risk of using COTS autcoders that lack a trusted heritage.

The AutoCert technology also has a number of advantages over other approaches to formal verification. It can handle code with arbitrary loops, and can handle code generated from both continuous and discrete models. Moreover, the certification system based on annotation inference is more flexible and extensible than decentralized architectures where certification information is distributed throughout the code generator. Identifying the patterns that are used to infer the annotations is an iterative process, but by allowing tracing between VCs (verification conditions) and state-