# DQG: Dual Quaternion Guidance for the SPLICE Project

**Taylor P. Reynolds**

Presented to: Blue Origin

February 10, 2021

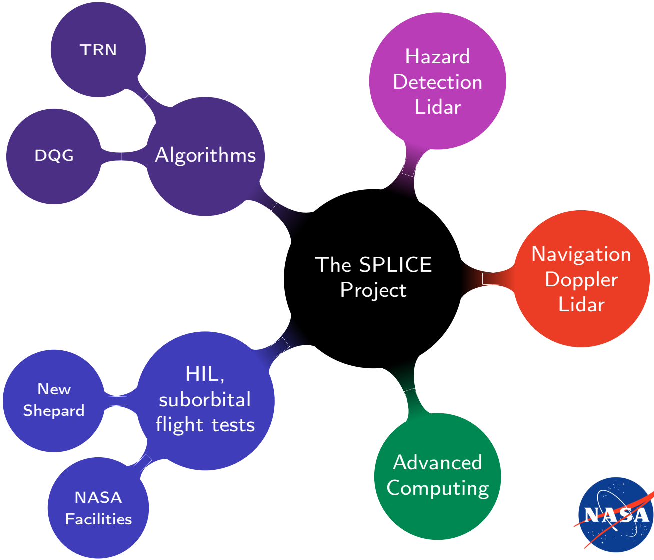
# Overview

# Overview

# Safe & Precise Landing - Integrated Capabilities Evolution

### Objectives

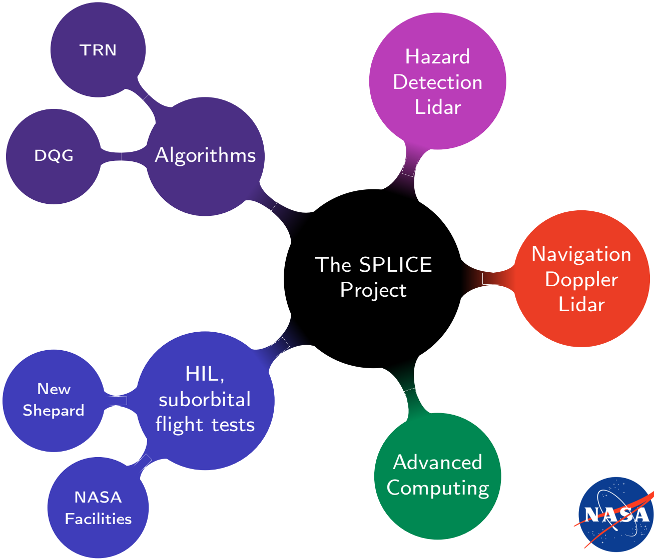
- Modernize precision landing technology
- Create target-agnostic solutions
- Achieve landing accuracy of $\mathcal{O}(10)$ m



# Safe & Precise Landing - Integrated Capabilities Evolution

**Timeline**

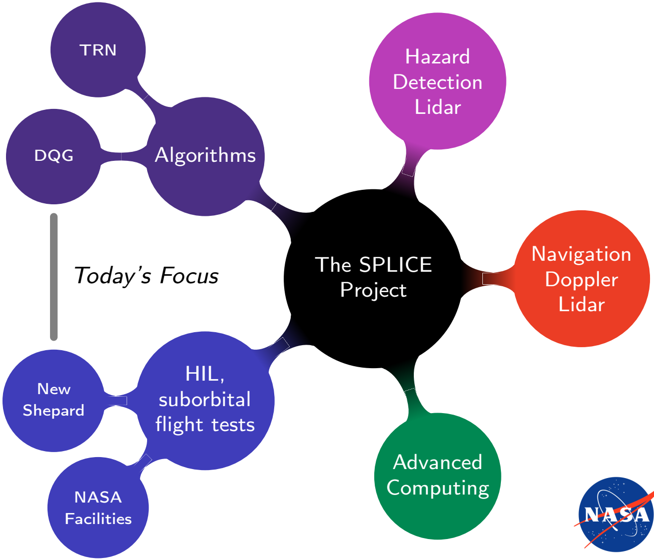
| | |
|---|---|
| 2005-2014 | ALHAT |
| 2015-2017 | COBALT |
| 2018 | SPLICE kick-off |
| 2018-2019 | ConOps, requirements analysis, tech. dev. |
| 2020-2021 | Prelim. flight demos, tech to TRL 5+ |
| 2022-2024 | CLPS demonstration |
| $\geq$ 2024 | Mission infusion |



# Safe & Precise Landing - Integrated Capabilities Evolution

**Collaborators**

- **JSC**[1], LaRC, GSFC, JPL, AFRC, MSFC
- Draper, Blue Origin, Coherent Applications, Masten Space Systems, ...
- UW, Texas A&M, ...




[1] All funding through NASA's STMD

# DQG Algorithm Development

**University of Washington:**

- Basic research on:
  - solution of nonconvex OCPs
  - problem formulations for lunar descent
- Develop Matlab-based toolbox to test and compare formulations and solution methods
- Develop flight code prototype in C/C++

**Team:**

TPR, M. Szmuk, D. Malyuta

M. Mesbahi, B. Açıkmeşe

**Draper Laboratory:**

- Develop cFS application for DQG
- Implement DQG prototype on SPLICE compute hardware

**Team:**

Javier Doll

Matt Fritz, Tim Barrows

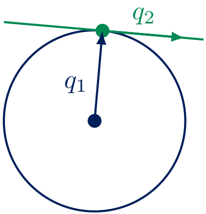
# Overview

# Dual Quaternions

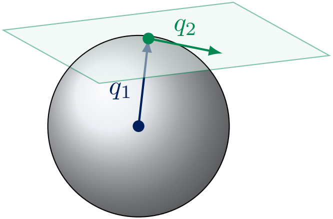
- A unit dual quaternion, $\tilde{q}$, satisfies two properties:

$$\tilde{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}_{8\times 1} \quad \text{where} \quad q_1^\top q_1 = 1, \quad \text{and} \quad q_1^\top q_2 = 0.$$

- DQG uses a right-handed, Hamiltonian, scalar-last convention
- The geometry of unit dual quaternions in lower dimensions:




dimension: $2$




dimension: $3$

?

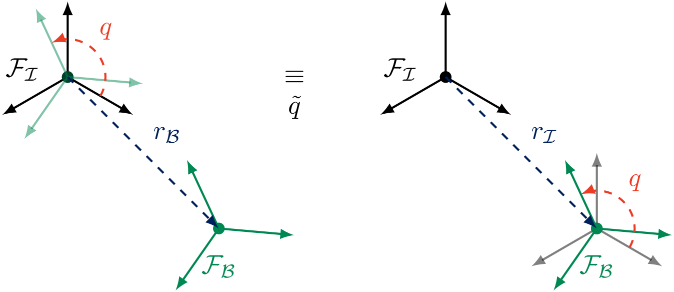
dimension: $4$

# Dual Quaternions & Rigid Body Motion

- The unit dual quaternion $\tilde{q}$ represents the pose of $\mathcal{F}_{\mathcal{B}}$ with respect to $\mathcal{F}_{\mathcal{I}}$:

$$\tilde{q} = \begin{bmatrix} q \\ \frac{1}{2} q \otimes r_{\mathcal{B}} \end{bmatrix} \equiv \begin{bmatrix} q \\ \frac{1}{2} r_{\mathcal{I}} \otimes q \end{bmatrix} \quad \text{and} \quad \tilde{\omega} = \begin{bmatrix} \omega_{\mathcal{B}} \\ v_{\mathcal{B}} \end{bmatrix}$$

- The dual velocity $\tilde{\omega}$ is composed of angular velocity and $v_{\mathcal{B}} = \dot{r}_{\mathcal{B}} + \omega_{\mathcal{B}}^{\times} r_{\mathcal{B}}$



# Optimal Control Problem

**Cost Function**

$$\max \quad m(t_f)$$ [C]

**Dual Quaternion Dynamics**

$$\dot{m} = -\alpha \|u_{\mathcal{B}}\|_2$$ [NC]

$$\dot{\tilde{q}} = \frac{1}{2}\tilde{q} \otimes \tilde{\omega}$$ [NC]

$$J\dot{\tilde{\omega}} = \Phi u_{\mathcal{B}} - \tilde{\omega} \oslash J\tilde{\omega} + m\tilde{g}_{\mathcal{B}}$$ [NC]

- 15-dimensional state vector (1+8+6)
- $u_{\mathcal{B}}$ is thrust vector, $g_{\mathcal{B}}$ is gravity
- matrix $\Phi$ maps thrust to a force & torque

# Optimal Control Problem

**Cost Function**

$$\max \quad m(t_f)$$ [C]

**Dual Quaternion Dynamics**

$$\dot{m} = -\alpha \|u_{\mathcal{B}}\|_2$$ [NC]

$$\dot{\tilde{q}} = \frac{1}{2}\tilde{q} \otimes \tilde{\omega}$$ [NC]

$$J\dot{\tilde{\omega}} = \Phi u_{\mathcal{B}} - \tilde{\omega} \oslash J\tilde{\omega} + m\tilde{g}_{\mathcal{B}}$$ [NC]

**Control Constraints**

$$u_{\min} \leq \|u_{\mathcal{B}}\|_2 \leq u_{\max}$$ [NC]

$$\|u_{\mathcal{B}}\|_2 \leq \sec\delta_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}}$$ [C]

$$-\dot{u}_{z,\max} \leq z_{\mathcal{B}}^\top u_{\mathcal{B}} \leq \dot{u}_{z,\max}$$ [C]

$$\|E_{xy} u_{\mathcal{B}}\|_2 \leq \dot{\delta}_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}}$$ [C]

- upper/lower throttle constraint
- gimbal angle constraint, $z_{\mathcal{B}}$ is vertical direction
- approx. throttle rate constraint
- approx. gimbal angle constraint

# Optimal Control Problem

**Cost Function**

$$\max \quad m(t_f) \qquad \text{[C]}$$

**Dual Quaternion Dynamics**

$$\dot{m} = -\alpha \|u_{\mathcal{B}}\|_2 \qquad \text{[NC]}$$

$$\dot{\tilde{q}} = \tfrac{1}{2}\tilde{q} \otimes \tilde{\omega} \qquad \text{[NC]}$$

$$J\dot{\tilde{\omega}} = \Phi u_{\mathcal{B}} - \tilde{\omega} \oslash J\tilde{\omega} + m\tilde{g}_{\mathcal{B}} \qquad \text{[NC]}$$

**Control Constraints**

$$u_{\min} \le \|u_{\mathcal{B}}\|_2 \le u_{\max} \qquad \text{[NC]}$$

$$\|u_{\mathcal{B}}\|_2 \le \sec\delta_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}} \qquad \text{[C]}$$

$$-\dot{u}_{z,\max} \le z_{\mathcal{B}}^\top u_{\mathcal{B}} \le \dot{u}_{z,\max} \qquad \text{[C]}$$

$$\|E_{xy} u_{\mathcal{B}}\|_2 \le \dot{\delta}_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}} \qquad \text{[C]}$$

**State Constraints**

$$-\tilde{q}^\top M_\gamma \tilde{q} + \|2E_d\tilde{q}\|_2 \cos\gamma_{\max} \le 0 \qquad \text{[C]}$$

$$\tilde{q}^\top M_\theta \tilde{q} + \cos\theta_{\max} \le 0 \qquad \text{[C]}$$

$$m_{dry} \le m \qquad \text{[C]}$$

$$\|E_v\tilde{\omega}\|_2 \le v_{\max} \qquad \text{[C]}$$

$$\|E_w\tilde{\omega}\|_\infty \le \omega_{\max} \qquad \text{[C]}$$

- approach angle (glide slope) and tilt angle expressed as quadratic functions of $\tilde{q}$
- mass, speed, and angular rate constraints

# Optimal Control Problem

## Cost Function

$$\max \quad m(t_f) \qquad \text{[C]}$$

## Dual Quaternion Dynamics

$$\dot{m} = -\alpha \|u_{\mathcal{B}}\|_2 \qquad \text{[NC]}$$

$$\dot{\tilde{q}} = \frac{1}{2}\tilde{q} \otimes \tilde{\omega} \qquad \text{[NC]}$$

$$J\dot{\tilde{\omega}} = \Phi u_{\mathcal{B}} - \tilde{\omega} \oslash J\tilde{\omega} + m\tilde{g}_{\mathcal{B}} \qquad \text{[NC]}$$

## State Constraints

$$-\tilde{q}^\top M_\gamma \tilde{q} + \|2E_d\tilde{q}\|_2 \cos\gamma_{\max} \le 0 \qquad \text{[C]}$$

$$\tilde{q}^\top M_\theta \tilde{q} + \cos\theta_{\max} \le 0 \qquad \text{[C]}$$

$$m_{dry} \le m \qquad \text{[C]}$$

$$\|E_v\tilde{\omega}\|_2 \le v_{\max} \qquad \text{[C]}$$

$$\|E_w\tilde{\omega}\|_\infty \le \omega_{\max} \qquad \text{[C]}$$

## Control Constraints

$$u_{\min} \le \|u_{\mathcal{B}}\|_2 \le u_{\max} \qquad \text{[NC]}$$

$$\|u_{\mathcal{B}}\|_2 \le \sec\delta_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}} \qquad \text{[C]}$$

$$-\dot{u}_{z,\max} \le z_{\mathcal{B}}^\top u_{\mathcal{B}} \le \dot{u}_{z,\max} \qquad \text{[C]}$$

$$\|E_{xy}u_{\mathcal{B}}\|_2 \le \dot{\delta}_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}} \qquad \text{[C]}$$

## Boundary Conditions

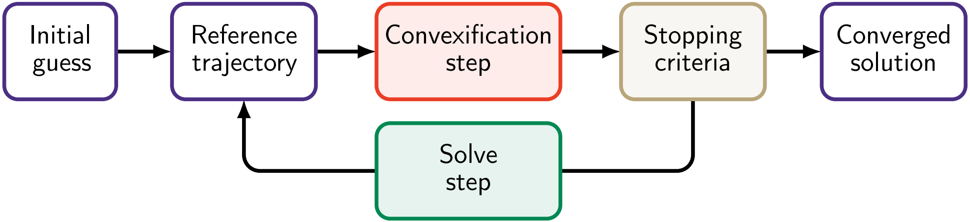
$$m(t_0) = m_{ic} \qquad \text{[C]}$$

$$\tilde{q}(t_0) = b_{\tilde{q}}\big(q(t_0)\big), \quad b_f\big(\tilde{q}(t_f)\big) \le 0 \qquad \text{[C]}$$

$$\tilde{\omega}(t_0) = b_{\tilde{\omega}}\big(q(t_0)\big), \quad \tilde{\omega}(t_f) = \tilde{\omega}_f \qquad \text{[NC]}$$

# Overview

# Sequential Convex Programming




- Solve nonconvex OCP by solving a sequence of convex approximations
- Initial guess obtained by simply interpolating between current and desired final states
- Converged solution satisfies dynamics to pre-determined precision

M. Szmuk et al. *JGCD*, 2020 | TPR et al. *JGCD*, 2020.

# Solve Step

- Each iteration solves a **s**econd-**o**rder **c**one **p**rogram using custom solver **BSOCP**[1]
- BSOCP is written in C++, but can generate C code tailored to the specific problem

## OCP

- convex cost function
- nonlinear, differential eq. constraints
- general ineq. and eq. constraints

$$\begin{aligned} \min_{u,p} \quad & J(x,u,p) \\ \text{s.t.} \quad & \dot{x} = f(x,u,p) \\ & 0 \geq g(x,u,p) \\ & 0 = h(x,u,p) \end{aligned}$$

## SOCP

- linear cost function
- affine, algebraic eq. constraints
- affine or second-order cone ineq. constraints

$$\begin{aligned} \min_{z} \quad & c^\top z \\ \text{s.t.} \quad & Az = b \\ & z \in \mathcal{C}_L \times \mathcal{C}_{Q_1} \times \cdots \times \mathcal{C}_{Q_m} \end{aligned}$$

[1] D. Dueri, 2018

# Solve Step

- Each iteration solves a **s**econd-**o**rder **c**one **p**rogram using custom solver **BSOCP**[1]
- BSOCP is written in C++, but can generate C code tailored to the specific problem

Convexification

**OCP**

- convex cost function
- nonlinear, differential eq. constraints
- general ineq. and eq. constraints

**SOCP**

- linear cost function
- affine, algebraic eq. constraints
- affine or second-order cone ineq. constraints

$$\begin{aligned} \min_{u,p} \quad & J(x,u,p) \\ \text{s.t.} \quad & \dot{x} = f(x,u,p) \\ & 0 \geq g(x,u,p) \\ & 0 = h(x,u,p) \end{aligned}$$

$$\begin{aligned} \min_{z} \quad & c^\top z \\ \text{s.t.} \quad & Az = b \\ & z \in \mathcal{C}_L \times \mathcal{C}_{Q_1} \times \cdots \times \mathcal{C}_{Q_m} \end{aligned}$$

[1] D. Dueri, 2018

# Solve Step

Solve step

- Each iteration solves a **s**econd-**o**rder **c**one **p**rogram using custom solver **BSOCP**[1]
- BSOCP is written in C++, but can generate C code tailored to the specific problem

**OCP**

- convex cost function
- nonlinear, differential eq. constraints
- general ineq. and eq. constraints

$$\begin{aligned} \min_{u,p} \quad & J(x,u,p) \\ \text{s.t.} \quad & \dot{x} = f(x,u,p) \\ & 0 \geq g(x,u,p) \\ & 0 = h(x,u,p) \end{aligned}$$

**SOCP**

- linear cost function
- affine, algebraic eq. constraints
- affine or second-order cone ineq. constraints

$$\begin{aligned} \min_{z} \quad & c^\top z \\ \text{s.t.} \quad & Az = b \\ & z \in \mathcal{C}_L \times \mathcal{C}_{Q_1} \times \cdots \times \mathcal{C}_{Q_m} \end{aligned}$$

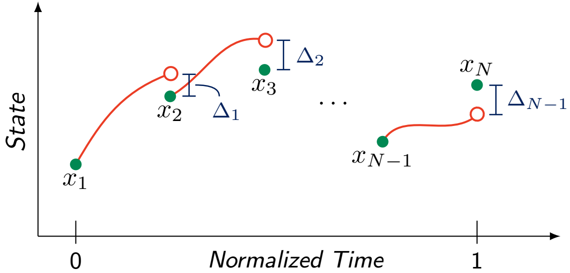
**Parsing**

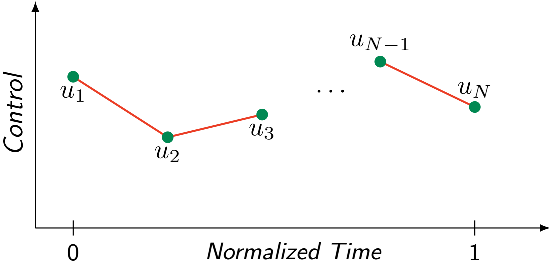
[1] D. Dueri, 2018

# Convexification: Propagation

Convexification step

- Assume the control can be affinely interpolated between a set of $N$ time nodes
- Use an exact discretization; solution of linearized ODE via numerical integration
- *Defects* $\Delta$ serve as an indicator of dynamic feasibility
- Properly implemented, this should take roughly $\mathcal{O}(1)\%$ of the DQG's runtime



# Convexification: Constraint Approximation

Convexification step

### Control Constraints

| | | |
|---|---|---|
| $u_{\min} \leq \|u_{\mathcal{B}}\|_2 \leq u_{\max}$ | [NC] | *(linearized + SOC)* |
| $\|u_{\mathcal{B}}\|_2 \leq \sec \delta_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}}$ | [C] | *(SOC)* |
| $-\dot{u}_{z,\max} \leq z_{\mathcal{B}}^\top u_{\mathcal{B}} \leq \dot{u}_{z,\max}$ | [C] | *(affine)* |
| $\|E_{xy} u_{\mathcal{B}}\|_2 \leq \dot{\delta}_{\max} z_{\mathcal{B}}^\top u_{\mathcal{B}}$ | [C] | *(SOC)* |

### State Constraints

| | | |
|---|---|---|
| $-\tilde{q}^\top M_\gamma \tilde{q} + \|2E_d\tilde{q}\|_2 \cos\gamma_{\max} \leq 0$ | [C] | *(linearized)* |
| $\tilde{q}^\top M_\theta \tilde{q} + \cos\theta_{\max} \leq 0$ | [C] | *(SOC)* |
| $m_{dry} \leq m$ | [C] | *(affine)* |
| $\|E_v \tilde{\omega}\|_2 \leq v_{\max}$ | [C] | *(SOC)* |
| $\|E_w \tilde{\omega}\|_\infty \leq \omega_{\max}$ | [C] | *(affine)* |

- Approach cone constraint is not an SOC, must linearize
- Max. violations of approx'd constraints tracked using parameter: $\delta$

- Each constraint enforced at $N$ nodes
- To parse SOCP: can quantify the min. number of variables and rows of $A$, $b$

# Convexification: Constraint Approximation

Convexification step

**Boundary Conditions**

At $t_0$ :

$m(t_0) = m_{ic}$ [C] *(affine)*

$\tilde{q}(t_0) = b_{\tilde{q}}\big(q(t_0)\big)$ [C] *(affine)*

$\tilde{\omega}(t_0) = b_{\tilde{\omega}}\big(q(t_0)\big)$ [NC] *(linearized)*

At $t_f$ :

$b_f\big(\tilde{q}(t_f)\big) \leq 0$ [C] *(SOC)*

$\tilde{\omega}(t_f) = \tilde{\omega}_f$ [C] *(affine)*

- Specified:
  - initial mass, (inertial) position and velocity, angular rates
  - final attitude, (inertial) velocity, angular rates

$$b_{\tilde{q}}\big(q(t_0)\big) = \begin{bmatrix} q(t_0) \\ \frac{1}{2} r_{\mathcal{I}} \otimes q(t_0) \end{bmatrix}$$

$$b_{\tilde{\omega}}\big(q(t_0)\big) = \begin{bmatrix} \omega_{\mathcal{B}}(t_0) \\ q(t_0)^* \otimes v_{\mathcal{I}}(t_0) \otimes q(t_0) \end{bmatrix}$$

$$b_f\big(\tilde{q}(t_f)\big) = \|2E_d\tilde{q}(t_f) - c'\|_2 - \varepsilon_{\text{miss}}$$

where $E_d = \mathbf{diag}\left\{0_{4\times 4}, I_4\right\}$

# Trust Regions and Virtual Control

Solve step

- Algorithmic modifications to ensure:
  - Each SOCP is feasible and bounded
  - Iterates kept "close" to the reference used for approximation
- **Virtual control** added to all approximated constraints:

$$\text{[NC]}: \quad h(x) = 0 \qquad \Longrightarrow \qquad \text{[C]}: \quad h(\bar{x}) + \nabla h(\bar{x})^\top \big(x - \bar{x}\big) + \nu = 0$$

  where $\nu$ is *unconstrained* but highly *penalized* in the cost.
- **Trust region** added as additional constraints:

$$\|x - \bar{x}\|_2^2 + \|u - \bar{u}\|_2^2 \leq \eta \quad \text{and} \quad \|p - \bar{p}\|_2^2 \leq \eta_p$$

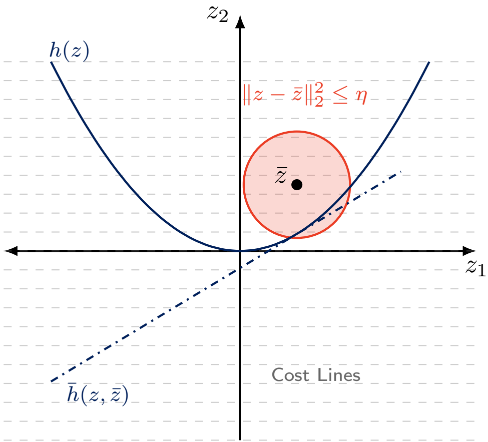
  where $\eta, \eta_p$ are chosen by the solver and modestly penalized in the cost.

**Cost Function**

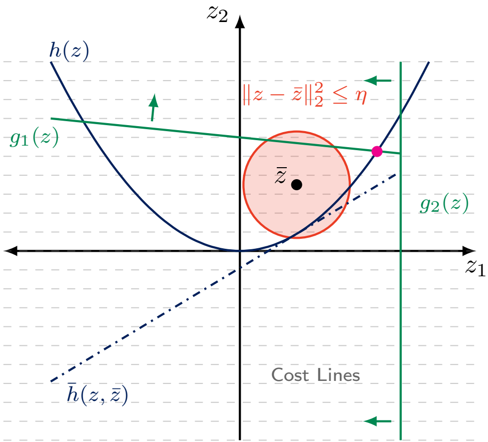
$$\max \quad m(t_f) \; - \; w_{tr}^\top \eta \; - \; w_{tr,p} \eta_p \; - \; w_{vc} \textstyle\sum_k \|\nu_k\|_1$$

# Trust Regions and Virtual Control

Solve step




**Artificial Unboundedness:** cost function is unbounded without a trust region




**Artificial Infeasibility:** no feasible solution with approximated constraint $\bar{h}$

# Stopping Criteria

**Criteria:**

1. small defects & constraint violation $\max\left\{\max_k \ \Delta_k,\, \delta\right\} \leq \varepsilon_\Delta$
2. small state change $\max_k \ \|x_k - \bar{x}_k\|_2 \leq \varepsilon_x$
3. small final mass change $|m_N - \bar{m}_N| \leq \varepsilon_m$

**Logic:**

1 AND ( 2 OR 3 )

- ➥ $\varepsilon_\Delta$ measures "feasibility", both dynamic and approx'd constraints
- ➥ $\varepsilon_m$ used to stop if optimality not sufficiently improving
- ➥ Control changes ignored: large thrust change can have small impact on state trajectory

# Overview

# Mission-Specific Initial Guess

- Currently, DQG uses a very simple initialization:

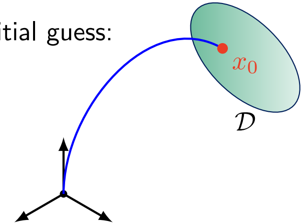
$$x_k = \left(\frac{N-k}{N-1}\right) x_{ic} + \left(\frac{k-1}{N-1}\right) x_{fc} \quad \text{and} \quad u_k = m_k g_{\mathcal{B}}, \quad k = 1, \ldots, N$$

  where $x_{ic}$ and $x_{fc}$ are boundary conditions, $g_{\mathcal{B}}$ is gravity in body frame $\mathcal{F}_{\mathcal{B}}$
- This trajectory is **not feasible** but works well enough and is easy to compute

Better initial guess $\Rightarrow$ fewer DQG iterations $\Rightarrow$ lower runtime

- **Idea:** For a given mission, create a map from initial condition to initial guess:

$$\begin{aligned} \Psi : \mathcal{D} &\to \mathcal{X} \times \mathcal{U} \times \mathbb{R}_{++} \\ x_0 &\mapsto \{x_k,\, u_k,\, t_f\}_{k=1}^{N} \end{aligned}$$



# Custom Conversion to Standard Form

- BSOCP solves a problem in *standard form*:

$$\begin{aligned} \min \quad & c^\top z \\ \text{s.t.} \quad & Az = b \\ & z \in \mathcal{C}_L \times \mathcal{C}_{Q_1} \times \cdots \times \mathcal{C}_{Q_m} \end{aligned}$$

- Two possible methods to define data $A$, $b$, $c$ and the cone dimensions:

**Generic Parser (Current)**

✓ Ideal for prototyping
✗ Can add variables/constraints
✗ Opaque problem construction
✗ Uses dynamic memory allocation

**Handparsing**

✗ Least flexible
✓ Guarantees smallest problem
✓ Most control over coding
✓ Uses static memory allocation

## Handparsing to Standard Form

$$z = \begin{bmatrix} \bigstar^z_1 \\ \bigstar^z_2 \\ \bigstar^z_3 \end{bmatrix} \begin{matrix} \} \text{ block } v_1 \\ \} \text{ block } v_2 \\ \} \text{ block } v_3 \end{matrix}$$

$$\begin{array}{cccc} & \text{block } v_1 & \text{block } v_2 & \text{block } v_3 \end{array}$$

$$A = \begin{bmatrix} \bigstar^A_P & 0 & 0 \\ \bigstar^A_C & \bigstar^A_s & \bigstar^A_\chi \end{bmatrix} \begin{matrix} \text{block } c_1 \\ \text{block } c_2 \end{matrix} \begin{bmatrix} \bigstar^b_P \\ \bigstar^b_C \end{bmatrix} = b$$

$$c = \begin{bmatrix} \bigstar^c_\phi & \bigstar^c_s & \bigstar^c_\chi \end{bmatrix}$$

- Variables:
  - $v_1$: used to write dynamics
  - $v_2$: linear slack variables
  - $v_3$: trust region and SOC slack variables
- Constraints:
  - $c_1$: dynamics
  - $c_2$: all other constraints

TPR et al. *SciTech*, 2020

# Handparsing to Standard Form

$$z = \begin{bmatrix} \bigstar_1^z \\ \bigstar_2^z \\ \bigstar_3^z \end{bmatrix} \begin{matrix} \}\ \text{block } v_1 \\ \}\ \text{block } v_2 \\ \}\ \text{block } v_3 \end{matrix}$$

$$\begin{array}{cccc} \text{block } v_1 & \text{block } v_2 & \text{block } v_3 & \\ \end{array}$$

$$A = \begin{bmatrix} \bigstar_P^A & 0 & 0 \\ \bigstar_C^A & \bigstar_s^A & \bigstar_\chi^A \end{bmatrix} \begin{matrix} \text{block } c_1 \\ \text{block } c_2 \end{matrix} \begin{bmatrix} \bigstar_P^b \\ \bigstar_C^b \end{bmatrix} = b$$

$$c = \begin{bmatrix} \bigstar_\phi^c & \bigstar_s^c & \bigstar_\chi^c \end{bmatrix}$$

- blocks $\bigstar$: can be pre-parsed
- blocks $\bigstar$: are updated each iteration

- Variables:
  - $v_1$: used to write dynamics
  - $v_2$: linear slack variables
  - $v_3$: trust region and SOC slack variables
- Constraints:
  - $c_1$: dynamics
  - $c_2$: all other constraints

TPR et al. *SciTech*, 2020

# State-Triggered Constraints (STC)

- STCs are constraints enforced conditionally based on the value of a *trigger function*

$$\underbrace{g(z) < 0}_{\textit{trigger condition}} \quad \Rightarrow \quad \underbrace{h(z) \leq 0}_{\textit{constraint condition}}$$

- Equivalently, we can enforce the nonconvex constraint

$$-\min\big(g(z),\, 0\big)\, h(z) \leq 0$$

- Models binary decisions using continuous variables

- Can combine trigger/constraint conditions using Boolean *AND* and *OR* operations

M. Szmuk et al. *JGCD*, 2020

# State-Triggered Constraints (STC)

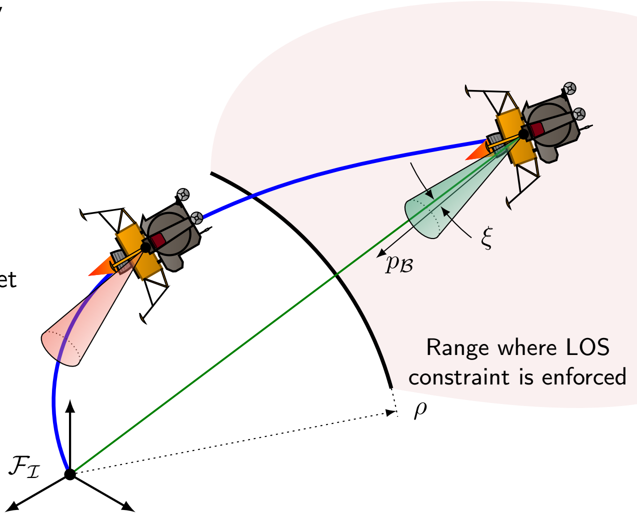
- Reconciles vehicle configuration with feasbility of optimal control problem

- Trigger: slant range larger than $\rho$

$$g(\tilde{q}) = \rho - \|2E_d\tilde{q}\|_2$$

- Constraint: line of sight angle to landing target

$$h(\tilde{q}) = \tilde{q}^\top M_\xi \tilde{q} + \|2E_d\tilde{q}\|_2 \cos\xi_{\max} - \varepsilon$$



# State-Triggered Constraints (STC)

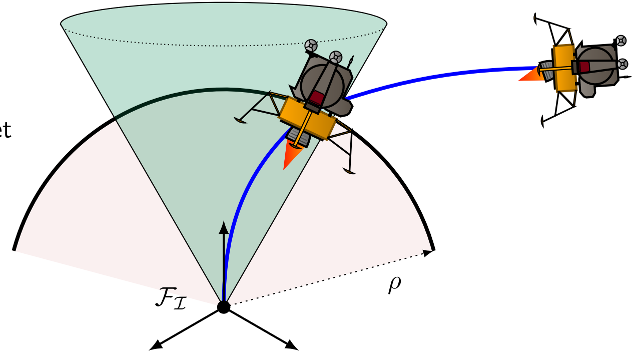
- Lunar descent orbits characteristically low altitude at large downrange distances

- Trigger: slant range small than $\rho$

$$g(\tilde{q}) = \|2E_d\tilde{q}\|_2 - \rho$$

- Constraint: approach angle to landing target

$$h(\tilde{q}) = -\tilde{q}^\top M_\gamma \tilde{q} + \|2E_d\tilde{q}\|_2 \cos\gamma_{\max}$$

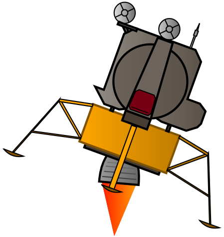


# Summary

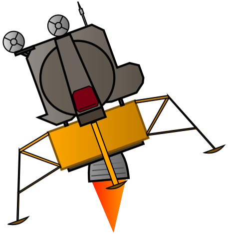
- DQG solves a nonconvex optimal control problem in real-time
  - parameterizes pose with dual quaternions
  - includes several 6-DOF constraints
  - uses sequential convex programming

- Part of NASAs SPLICE project to modernize autonomous precision landing
  - algorithm dev: UW, Draper, JSC
  - HIL/flight demo: Blue Origin, Draper, JSC

- Plenty of room for improvement, both research-based and on the DQG implementation



# Thank you!



email: tpreynolds6@gmail.com

| **UW** | **JSC** | **Draper** |
|---|---|---|
| M. Szmuk | J. M. Carson III | J. Doll |
| D. Malyuta | R. Sostaric | M. Fritz |
| M. Mesbahi | D. Matz | T. Barrows |
| B. Açıkmeşe | E. Braden | R. Loffi |
| U. Lee | | |

# A Few Relevant Publications


**Algorithm Development**

- TPR, M. Szmuk, D. Malyuta, M. Mesbahi, B. Açıkmeşe and J. M. Carson III, "Dual Quaternion Based Powered Descent Guidance with State-Triggered Constraints," *J. of Guidance, Control and Dynamics*, vol. 43, no. 9, pp. 1584-1599, 2020
- M. Szmuk, TPR, and B. Açıkmeşe, "Successive Convexification for Real-Time 6-DoF Powered Descent Guidance with State-Triggered Constraints," *J. of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1399-1413, 2020
- TPR, D. Malyuta, M. Mesbahi, B. Açıkmeşe and J. M. Carson III, "A Real-Time Algorithm for Non-Convex Powered Descent Guidance," *AIAA SciTech Forum*, Orlando, FL. 2020
- D. Malyuta, TPR, M. Szmuk, M. Mesbahi, B. Açıkmeşe, and J. M. Carson III, "Discretization Performance and Accuracy Analysis for the Powered Descent Guidance Problem," *AIAA SciTech Forum*, Orlando, FL. 2019

**SPLICE**

- R. Sostaric, S. Pedrotty, J. M. Carson III, et al., "The SPLICE Project: Safe and Precise Landing Technology Development and Testing," *AIAA SciTech Forum*, Virtual, 2021
- J. M. Carson III, M. M. Munk, R. Sostaric, et al., "The SPLICE Project: Continuing NASA Development of GN&C Technologies for Safe and Precise Landing," *AIAA SciTech Forum*, Orlando, FL. 2020