2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Gasoline Controlled Auto-Ignition with Learning-based Uncertainty Using Stochastic Model Predictive Control

Xu Chen, Maximilian Basler, Maike Stemmler, Dirk Abel

Abstract— **The internal combustion engine faces a severe energy conservation and emission reduction challenge. In this regard, low-temperature combustion technology is a profitable solution, allowing for pollutant emission reduction while improving engine efficiency. However, the process is complex, and the cycles are mutually coupled, making it a huge challenge to stabilize the entire process behavior. Also, the model mismatch and the inherent stochasticity of the process bring considerable difficulties to the application of control technology. In this work, we propose a deep learning-based generative model to learn the distribution of system uncertainties. The uncertainty information is considered in the model predictive control (MPC) strategy design. We adopt the disturbance-affine stochastic MPC (sMPC) and transform the chance-constrained MPC problem into some tractable optimization problems. The results show better closed-loop performance with smaller output variance, given the prior knowledge of uncertainty realization from the proposed generative model.**

I. INTRODUCTION

Gasoline controlled auto-ignition (GCAI), a form of low-temperature combustion technology, can potentially reduce CO_2 emissions and significantly lower NO_x raw emissions [1]. The combustion process is driven by chemical chain reactions and characterized by a homogeneous mixture. Various techniques have been developed to actively influence the combustion process, such as adjusting the air intake quantity, mixing fuels with different reactants, and recirculating the burnt residual mixture. One of the most common and practical strategies is exhaust gas recirculation, achieved by closing the exhaust valve early [2].

Since GCAI combustion induced by compression cannot be directly controlled as ignition in spark ignition engines or injection in compression ignition engines, the process is highly dependent on the thermodynamic state and composition of the combustion mixture, which limits the operating range of GCAI combustion [3]. Near the stable operating boundary, stochastic cycles with irregular combustion, which cannot be accurately predicted, might occur. When the residual gas of the combustion outlier transfers into the subsequent cycle by gas recirculation, a strong coupling between the combustion cycles will be caused, leading to unstable operating stages and strong cyclic variations. Thus, controlling GCAI combustion is a challenging task [4].

Over the past decades, many combustion models for simulation purposes and controller design have been the focus of research. Among them, physics-based modeling has made significant progress, such as zero-dimensional single-zone combustion and multi-zone models [5]. However, most are physically detailed and overly complex, which increases development costs and computational requirements. It leads to difficulties in real-time implementation.

Recently, with the development in hardware capabilities and the availability of large experimental data, machine learning (ML)-based methods have been developed [6]–[8]. These models can quickly learn the hidden mappings between inputs and measurements, considerably simplify the interactions between physical variables and provide the possibility of real-time implementation for model-based controllers. However, ML-based models trained with a large amount of data are robust to outliers in the training set, which makes it difficult to predict the stochastic combustion irregularities that might occur. Besides, predicting irregular combustions is also a big challenge for white-box models.

To handle the combustion outliers and other uncertainties (model mismatch, unknown disturbances, measurement noise, etc.), we propose a deep learning (DL)-based generative model method. Specifically, a DL-based nominal model is deployed to describe the system dynamics, and an additional model is utilized to approximate the uncertainty distribution. Unlike the previous methods using discriminative ML-based models, which can not fit the distribution as it is an unsupervised learning problem, we use generative models [9], [10] to learn the approximate uncertainty distribution and generate new uncertainty samples with certain variations. The generative models adopted in this work are flow-based models [11], [12], aiming to explicitly learn the probability distribution of the uncertainty by transforming the defined latent random variables. The parametrized generative model then generates the uncertainty samples. Given the uncertainty information, we apply sMPC to ensure the robustness to uncertainty in closed-loop performance. This work considers the MPC strategy with affine dependence on disturbance [13]–[15] and uses time-varying feedback gains instead of the fixed gains in the open-loop optimization strategy. Additionally, we transform the original optimization problem with chance constraints into some tractable problems.

The remainder of this work is organized as follows. Section 2 describes the experimental setup and the overview of the GCAI process. In Section 3, the flow-based generative model structures and the sMPC strategy are introduced. Section 4 presents the simulative results of the generative model and controller implementation. Finally, Section 5 concludes with the performance of the proposed method and future research directions.

Xu Chen, Maximilian Basler, Maike Stemmler and Dirk Abel are with Institute of Automatic Control, Department of Mechanical Engineering, RWTH Aachen University, 52074 Aachen, Germany x.chen@irt.rwth-aachen.de

II. Experimental setup

The size and quality of the training datasets are crucial factors in determining the performance of deep neural networks. Generally, larger datasets can effectively reduce the generalization error of models. For this study, we collected experimental data using the Single Cylinder Research Engine with a fully variable Electro-Magnetic Valve Train (EMVT).

The pressure trace concerning the crank angle in Fig. 1 depicts the typical GCAI process with the intermediate compression and the main combustion. The EMVT allows various valve strategies to operate the engine. In this work, we use the symmetric Negative Valve Overlap (NVO) strategy [18], which ensures that no intake or exhaust rebreathing occurs by varying Exhaust Valve Closing (EVC) and Intake Valve Opening (IVO) evenly around Top Dead Center (TDC). We can modify the NVO duration in every cycle for control purposes while Exhaust Valve Opening (EVO) and Intake Valve Closing (IVC) are constants. Besides, the amount of injected fuel and water significantly impacts the entire process. Therefore, in addition to NVO, we utilize the durations of fuel and water injection as control inputs, yielding three control inputs of the system.

As system outputs, we choose surrogate parameters that represent the cyclic combustion performance, namely combustion phasing (CA50), indicated mean effective pressure (IMEP), and maximum pressure rise rate (MPRR). CA50 represents the angle at which 50% of the heat is released. It is related to combustion efficiency and stability. IMEP is associated with the load, and MPRR is related to noise and mechanical stress. In this work, the primary uncertainty considerations concern CA50 and IMEP. For MPRR, we limit it below its upper limit with 5 bar/°CA. In addition, the experimental dataset was collected with various manipulated variables (NVO and injection durations) at the specified speed operating point of 1500 rpm.

III. Methodology

The GCAI combustion process is complicated, and the irregularity of combustion makes it difficult for the model to accurately predict the states of the next cycle. In this work, we assume that uncertainty is a random variable that follows a specific probability distribution. Then, we can consider the following discrete-time nonlinear system:

$$x_{k+1} = f(x_k, u_k) + g(x_k, w_k) \tag{1}$$

where $x_k \in \mathbb{R}^{n_x}$ is the state vector at the time instant k, $u_k \in \mathbb{R}^{n_u}$ is the control input vector and $w_k \in \mathbb{R}^{n_w}$ is the disturbance vector. The mapping $f\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ denotes the nominal nonlinear system, and $g\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ denotes the uncertainty model that describes the uncertainty realization. Here, we assume that w_k follows a standard normal distribution, which means that elements are mutually independent.

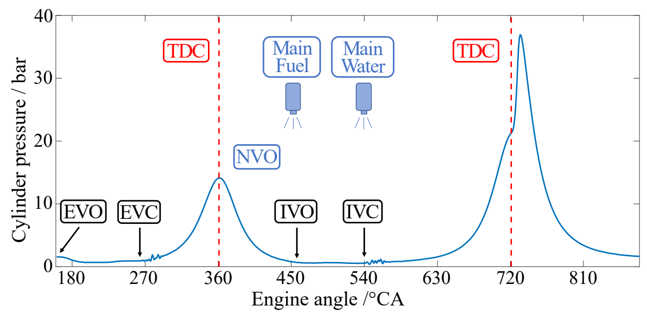

Fig. 1. GCAI pressure trace at CA50 = 6 °CA aTDC and IMEP = 2.9 bar.

A. Flow-based model structure

The uncertainty model g in (1) is related to the current states and disturbances, and its outputs, the uncertainties, are the observed variables. We aim to learn the distribution of the uncertainties based on a flow-based model [11], [12].

By introducing latent variables Z, which follow a standard normal distribution with the same dimensions as the observed variables (uncertainties) Y, and a fixed model structure with parameter θ, we can find the approximate marginal distribution of Y through the model parameterization. Specifically, we can find an approximate distribution q_θ (briefly q) to fit the real distribution p_Y and obtain samples of Y by sampling from the latent distribution of Z. The goal is to maximize the following objective function:

$$\mathbb{E}_{y \sim p_Y(y)} [\log q(y)] \tag{2}$$

Generally, we need to consider solving the following integration:

$$q(y) = \int_z q(z) q(y|z) dz \tag{3}$$

In other words, we need to solve the maximum log-likelihood problem with latent variables given in (2) and (3).

Commonly used generative models, like Variational Autoencoder (VAE) [9] and Generative Adversarial Networks (GAN) [10], use different strategies to tackle such problems. VAE utilizes an encoder-decoder structure to approximate the posterior probability of latent variables and optimize the lower bound of the likelihood. GANs solve the above problem through the adversarial interaction between a generator and a discriminator.

Unlike previously mentioned generative models, flow-based models used in this work avoid the integration in (3) by explicitly transferring the latent variables. The main idea of the models is to find a mapping h from observed variables to latent variables. In our hypothesis, mapping h additionally requires decoupling the effects of the system states on observed variables and ultimately transforming them into independent latent variables. Thus, our goal is to find the conditional probability distribution of observed variables given the current states.

We first focus on the latent variables Z and the observed variables Y. Specifically, we search for a variable transformation that ensures a bijective mapping h between two linear spaces with the same dimensions. In particular, if Y follows a zero-mean normal distribution, we can find an isomorphism between these two spaces, i.e., g in (1) can be presented as Dw_k with $D \in \mathbb{R}^{n_x \times n_w}$. The transformation for general cases can be written as:

$$p_Y(y) = p_Z(h(y)) \left|\det \frac{\partial h(y)}{\partial y}\right| \tag{4}$$

where p_Y is the probability density function (pdf) of Y, p_Z is the pdf of Z. The latter is the determinant of the Jacobian matrix, which can be the ratio of the differential volume element before and after the transformation. The transformation in (4) is a change of variable method obtained by differentiating the cumulative distribution functions on both sides. Meanwhile, the requirement on non-negativity and normalization of probability distributions is ensured. In this case, we obtain samples from the target variable distribution by changing the latent variable samples. Subsequently, we can find the corresponding mapping by maximizing the log-likelihood function below:

$$\log p_Y(y) = \log p_Z(h(y)) + \log \left|\det \frac{\partial h(y)}{\partial y}\right| \tag{5}$$

The question is how to define the mapping h or how to design the model so that h is bijective and the computation of the Jacobian matrix is easy. Here, we adopt a model similar to the structure in [11], which transforms the latent variables to the observed variables using a combination of coupling layers $h_0^{-1} \circ h_1^{-1} \circ \cdots \circ h_{n-1}^{-1}$. The number of layers n depends on the complexity of the distribution. The Jacobian determinant of the entire transformation can be obtained as the product of the Jacobian determinants of each layer. The definition of the i-th coupling layer is given by:

$$z_{I_1}^{(i+1)} = z_{I_1}^{(i)} \tag{6}$$

$$z_{I_2}^{(i+1)} = \tilde{g}\left(z_{I_2}^{(i)}, m\left(z_{I_1}^{(i)}\right)\right) \tag{7}$$

where the input $z^{(i)} \in \mathbb{R}^{d_1+d_2}$ is divided into $z_{I_1}^{(i)} \in \mathbb{R}^{d_1}$ and $z_{I_2}^{(i)} \in \mathbb{R}^{d_2}$, $m\colon \mathbb{R}^{d_1} \to \mathbb{R}^{n_m}$ is a nonlinear function of $z_{I_1}^{(i)}$, $\tilde{g}\colon \mathbb{R}^{d_2} \times \mathbb{R}^{n_m} \to \mathbb{R}^{d_2}$ is the coupling function, and $z^{(i+1)}$ is the i-th layer output. We use a simple neural network as the structure of m, which will be parameterized during training. Moreover, we choose the identity mapping in (6) and adopt an additive coupling in (7), which can also be other coupling schemes, such as multiplication coupling and affine coupling. In this case, (7) can be written as:

$$z_{I_2}^{(i+1)} = z_{I_2}^{(i)} + m\left(z_{I_1}^{(i)}\right) \tag{8}$$

and the Jacobian matrix of the coupling layer is given by:

$$\frac{\partial z^{(i+1)}}{\partial z^{(i)}} = \begin{bmatrix} I_{d_1} & 0 \\ \frac{\partial z_{I_2}^{(i+1)}}{\partial z_{I_1}^{(i)}} & I_{d_2} \end{bmatrix} \tag{9}$$

The determinant of this Jacobian matrix is simply one. However, this is the determinant of the inverse transformation, and we still need to invert it to keep consistent with the determinant in (5). Besides, we can obtain the inverse function of the coupling as:

$$z_{I_1}^{(i)} = z_{I_1}^{(i+1)} \tag{10}$$

$$z_{I_2}^{(i)} = z_{I_2}^{(i+1)} - m\left(z_{I_1}^{(i+1)}\right) \tag{11}$$

However, it is not sufficient to define a single layer with (6) and (8) for transformation due to identity mapping. We also need to exchange the order of coupling, i.e., (8) undergoes identity mapping, and (6) undergoes nonlinear mapping. According to this, more complex fitting effects can be achieved with increasing coupling layers.

Here, another issue is that the determinant of the Jacobian matrix for the entire model is always one, meaning that the differential volume element has not been changed. It may limit the network's capacity to some extent. To address this issue, we increase the degrees of freedom by introducing trainable scaling parameters to each coupling layer. Meanwhile, we also consider the influence of the current states on the nonlinear mapping m_i for each layer i. The final proposed model is shown in Fig. 2, where α_i denotes the trainable scaling in the layer i.

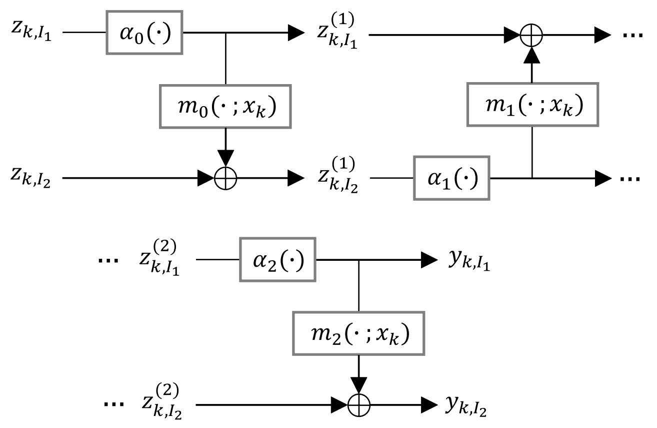

Fig. 2. The proposed flow-based model with three coupling layers. At each time instant k, z_k and x_k form the model inputs. In each layer i, nonlinear mapping m_i and scaling parameter α_i are applied, and $z_k^{(i+1)}$ is the layer output. The final output of the model is y_k.

By training the model with experimental datasets, including the current states and the prediction uncertainties, we can approximate the conditional distribution of the uncertainty vector given the states. In this case, g in (1) describes the proposed uncertainty model, and w is the latent variable following a standard normal distribution.

B. Stochastic model predictive control

In this subsection, we discuss the application of the generative model with sMPC for each combustion cycle. At the end of each cycle, we obtain the corresponding states, and the generative model provides prior knowledge about uncertainties to the MPC. In addition, we assume that a linearization of the model around the operation point is a sufficient good approximation. For convenient and effective use of control, we linearize the nominal model around the operating point for each cycle and the generative model around the current states and mean value of the latent variables. The linearized model is given by:

$$x_{k+1} = Ax_k + Bu_k + \tilde{D}(\bar{w})x_k + Dw_k \tag{12}$$

where $A \in \mathbb{R}^{n_x \times n_x}$ and $B \in \mathbb{R}^{n_x \times n_u}$ are the state and input matrix obtained by linearizing the nominal model, $\tilde{D} \in \mathbb{R}^{n_x \times n_x}$ is the Jacobian matrix of g in (1) with respect to x_k, and $D \in \mathbb{R}^{n_x \times n_w}$ is the Jacobian matrix of g with respect to w_k. Besides, $\bar{w}$ is the mean value of w_k.

As the disturbance vector w_k is a random variable and the parameters of $\tilde{D}$ change with the latent variable w_k, we can further linearize the matrix parameters to:

$$x_{k+1} = A^{(0)} x_k + Bu_k + \sum_{i=1}^{n_w} w_k^{(i)} A^{(i)} x_k + Dw_k \tag{13}$$

where $A^{(0)} = A + \tilde{D}$, the scalar $w_k^{(i)}$ represents the i-th element of w_k, and $A^{(i)} \in \mathbb{R}^{n_x \times n_x}$ is the uncertainty matrix concerning $w_k^{(i)}$. Then, (13) presents a system with mixed additive and multiplicative uncertainty. The bias produced by the generative model will be additionally added to the absolute prediction value. For simplicity, we only consider the linear system with additive uncertainty as shown in (12). Note that the additive uncertainty depends on the current states and changes as the state variables vary.

Now we consider the discrete-time linear system:

$$x_{k+1} = Ax_k + Bu_k + Dw_k \tag{14}$$

which is subject to the linear state constraints:

$$\mathcal{X} = \{x \in \mathbb{R}^{n_x} | Fx \leq f\} \tag{15}$$

where $F \in \mathbb{R}^{n_f \times n_x}$, $f \in \mathbb{R}^{n_f}$, and the inequality operation here is element-wise. The system is also subject to the linear input constraints:

$$\mathcal{U} = \{u \in \mathbb{R}^{n_u} | Gu \leq g\} \tag{16}$$

where $G \in \mathbb{R}^{n_g \times n_u}$, $g \in \mathbb{R}^{n_g}$. Next, we can minimize the following MPC problem:

$$\begin{aligned} & J(x_k, \mathbf{u}_k, \mathbf{w}_k) = \\ & \sum_{i=0}^{N-1} \left(\left\| \phi_{i|k}(x_k, \mathbf{u}_k, \mathbf{0}) - x_{ref} \right\|_Q^2 + \left\| u_{i|k}(x_k, \mathbf{0}) \right\|_R^2 \right) \\ & + \left\| \phi_{N|k}(x_k, \mathbf{u}_k, \mathbf{0}) \right\|_W^2 \\ s.t.\ \ & P(F\phi_{i|k}(x_k, \mathbf{u}_k, \mathbf{w}_k) \leq f) > p \quad \forall i \in \{0, ..., N\} \\ & P(Gu_{i|k}(x_k, \mathbf{w}_k) \leq g) > p \quad \forall i \in \{0, ..., N-1\} \\ & \mathbf{w}_k \sim \mathcal{N}(0_{n_w N}, I_{n_w N \times n_w N}) \end{aligned} \tag{17}$$

where x_k is the given state vector at the instant k, x_{ref} is the setpoint, $\mathbf{u}_k = \left[u_{0|k}^T \cdots u_{N-1|k}^T\right]^T$ is the predicted input sequence, $\mathbf{w}_k = \left[w_{0|k}^T \cdots w_{N-1|k}^T\right]^T$ is the predicted disturbance sequence, N is the prediction horizon, p is the probability of the constraint satisfaction, and $\phi_{i|k}$ is the solution to (14) at predicted time instant i. Here, we adopt the nominal cost without disturbance. As w_k follows a normal distribution, i.e., its support is unbounded, the state and input constraints must be represented as chance constraints.

Instead of commonly used open-loop control strategies with deterministic pre-stabilization feedback, we adopt a less restrictive control strategy, i.e., we utilize time-varying linear feedback combined with feedforward control. Here, the feedback gain $\mathbf{K}_k$ also depends on the evolution of the states, which can be written as:

$$\mathbf{u}_k = \mathbf{K}_k \mathbf{x}_k + \mathbf{v}_k \tag{18}$$

$$\mathbf{K}_k = \begin{bmatrix} K_{0,0|k} & 0 & \cdots & 0 & 0 \\ K_{0,1|k} & K_{1,1|k} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ K_{0,N-1|k} & K_{1,N-1|k} & \cdots & K_{N-1,N-1|k} & 0 \end{bmatrix} \tag{19}$$

where $\mathbf{x}_k = \left[x_{0|k}^T \cdots x_{N|k}^T\right]^T$ is the predicted state sequence with $x_{0|k} = x_k$, $\mathbf{v}_k = \left[v_{0|k}^T \cdots v_{N-1|k}^T\right]^T$ is the feedforward control sequence over the prediction horizon. Besides, we can write $\mathbf{x}_k$ in terms of x_k, $\mathbf{u}_k$ and $\mathbf{w}_k$ as:

$$\mathbf{x}_k = H_x x_k + H_u \mathbf{u}_k + H_w \mathbf{w}_k \tag{20}$$

with matrices H_x, H_u and H_w, which can be found in the appendix. By substituting (20) into (18), we can obtain the equation of $\mathbf{u}_k$ with respect to x_k, $\mathbf{w}_k$, $\mathbf{v}_k$, and $\mathbf{K}_k$, where $\mathbf{v}_k$ and $\mathbf{K}_k$ are optimization variables, and x_k is given at each time instant.

However, it is not a linear equation with respect to the optimization variables, which leads to a non-convex optimization problem. The good news is that we can convert the original formula into a linear equation through a bijective variable transformation via Youla parameterization [16]:

$$\mathbf{u}_k = \mathbf{V}_k \mathbf{w}_k + \mathbf{h}_k \tag{21}$$

with

$$\mathbf{h}_k = (I - \mathbf{K}_k H_u)^{-1} \mathbf{K}_k H_x x_k + (I - \mathbf{K}_k H_u)^{-1} \mathbf{v}_k \tag{22}$$

$$\begin{aligned} \mathbf{V}_k &= (I - \mathbf{K}_k H_u)^{-1} \mathbf{K}_k H_w \\ &= \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ V_{0,1|k} & 0 & \cdots & 0 & 0 \\ V_{0,2|k} & V_{1,2|k} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ V_{0,N-1|k} & V_{1,N-1|k} & \cdots & V_{N-2,N-1|k} & 0 \end{bmatrix} \end{aligned} \tag{23}$$

It is equivalent to (18) but with new variables $\mathbf{h}_k$ and $\mathbf{V}_k$. Both the new and old optimization variables can be converted into each other. Then, the predicted state sequence in (20) can be rewritten with the new variables as:

$$\mathbf{x}_k = H_x x_k + H_u \mathbf{h}_k + H_u \mathbf{V}_k \mathbf{w}_k + H_w \mathbf{w}_k \tag{24}$$

Next, we consider the expression of the chance constraints. The state chance constraints can be written in terms of (15) as:

$$P(\Omega) = \int_{\mathbf{W}} \mathbf{1}_{\Omega}(\mathbf{w}_k) dP = \int_{\Omega} p(\mathbf{w}_k) d\mathbf{w}_k > 1 - \epsilon \tag{25}$$

where P is a probability measure defined on the space $\mathbf{W}$ of $\mathbf{w}_k$ and the σ-algebra on $\mathbf{W}$, and p is the pdf of $\mathbf{w}_k$. Besides, the function $\mathbf{1}$ is the indicator function concerning the set Ω, while Ω is the feasible set where the constraints hold and is a measurable set. Moreover, ϵ is the probability of accepting constraint violations. The feasible set Ω is defined as:

$$\Omega = \{\mathbf{w}_k | \mathbf{F}\mathbf{x}_k - \mathbf{f} \leq 0\} \tag{26}$$

where $\mathbf{F} = I_{N+1\times N+1} \otimes F$, $\mathbf{f} = 1_{N+1\times 1} \otimes f$, and $\otimes$ is Kronecker product.

In (25), the joint probability of all constraints over the prediction horizon should be greater than the value of $1 - \epsilon$. It is easier to handle each constraint probability separately. Thus, we consider that the probability of being in the feasible set is $1 - \epsilon_0$ for each constraint, as follows:

$$P(\{\mathbf{w}_k | \mathbf{F}_j \mathbf{x}_k - \mathbf{f}_j \leq 0\}) > 1 - \epsilon_0 \tag{27}$$

where j denotes the j-th row of the matrix. Analog to the state constraints, we obtain the constraints for inputs as:

$$P(\{\mathbf{w}_k | \mathbf{G}_j \mathbf{u}_k - \mathbf{g}_j \leq 0\}) > 1 - \epsilon_1 \tag{28}$$

where ϵ_1 is the probability of the input constraint violation.

Meanwhile, we transform the optimization variables into vector form. Further explanations can be found in the appendix. Then, the state constraint inequality is written as:

$$F_{xj} x_k + F_{uj} \mathbf{h}_k + \mathbf{s}_k^T T_j^T \mathbf{w}_k + F_{wj} \mathbf{w}_k - \mathbf{f}_j \leq 0 \tag{29}$$

where $F_{xj} = \mathbf{F}_j H_x$, $F_{uj} = \mathbf{F}_j H_u$, $F_{wj} = \mathbf{F}_j H_w$, $T_j \in \mathbb{R}^{n_w N \times n_w n_u \frac{N(N-1)}{2}}$, $\mathbf{h}_k$ and $\mathbf{s}_k \in \mathbb{R}^{n_w n_u \frac{N(N-1)}{2}}$ are optimization variables.

To address the chance-constrained problem, we need to further transform the chance constraints. From (27), for any $\mathbf{w}_k$ that satisfies the normal distribution, we can find a feasible neighborhood ball $B(\mathbf{0}; r)$ around the mean point with radius r in the space of $\mathbf{w}_k$ so that the measure on this ball is a predefined value less than one. This allows us to obtain an uncertainty boundary for disturbance $\mathbf{w}_k$, which can be determined by the chi-square distribution. The chi-squared distribution represents the sum of squares of n independent standard normal random variables. For instance, the probability of a two-dimensional standard normal distribution within a circle $B(\mathbf{0}; r)$ can be expressed by this distribution. The degree of freedom n of the distribution is two. Then, the corresponding boundary r of the circle with a cumulative probability of 95%, as an example, is 2.45.

For any feasible optimization variables, the left side of the inequality in (29) is an affine function of $\mathbf{w}_k$, and the constraint forms a half-space with respect to $\mathbf{w}_k$, so the neighborhood ball $B(\mathbf{0}; r)$ must lie in this half-space. Then, the feasible set of optimization variables comprises the feasible optimization variables described above.

Besides, another two aspects need to be noted: 1. Only considering the measure on the neighborhood ball might be conservative. In any feasible half-space with fixed optimization variables, not just the ball, there exists a half-space passing through the mean point with a probability measure of 0.5. We should take the measures of both into account; 2. The constraint is only related to the realizations of the previous i disturbances at the i-th step of the prediction.

So far, we have transformed the chance constraint problem into a strict boundary problem. For simplicity, we consider expressing the boundary as the inscribed cube of the hypersphere. For instance, the unit circle in a two-dimensional space is expressed as the convex hull of the vertices of the inscribed square, i.e., $\text{conv}(\{(0,1), (0,-1), (1,0), (-1,0)\})$. In the region where the circle and the square do not intersect, the probability density is low, which is a reasonable assumption in low-dimensional space. But strictly considered, when the dimension is extremely high, the volume of the inscribed hypercube will decrease to zero as the dimension increases. Therefore, the assumption in extremely high-dimensional space is not a rational choice. This work focuses on incorporating flow-based models to learn the distribution of uncertainty. The purpose of sMPC is currently for convenience of application.

The simplified uncertainty set can be represented by a convex hull of the vertices, which we denote as $\mathcal{W}$. Then, (27) combined with (29) can be reformulated as:

$$F_{xj} x_k + F_{uj} \mathbf{h}_k + \max_{\mathbf{w}_k \in \mathcal{W}} \mathbf{s}_k^T T_j^T \mathbf{w}_k + F_{wj} \mathbf{w}_k - \mathbf{f}_j \leq 0 \tag{30}$$

If all vertices in set $\mathcal{W}$ satisfy the inequality condition in (30), the convex combination of the vertices will satisfy the condition. Thus, the constraints can be transformed into the following inequality constraints:

$$F_{xj} x_k + F_{uj} \mathbf{h}_k + \mathbf{s}_k^T T_j^T \mathbf{w}_k^{(n)} + F_{wj} \mathbf{w}_k^{(n)} - \mathbf{f}_j \leq 0 \quad \forall n \in \mathcal{N} \tag{31}$$

where $\mathbf{w}_k^{(n)}$ is the vertex of $\mathcal{W}$, and $\mathcal{N}$ is the index set of vertices of $\mathcal{W}$. Besides, the handling of input chance constraints is similar. As a result, the sMPC problem in (17) is transformed into a quadratic program optimization problem with constraints in (31). Note that the number of inequality constraints increases exponentially with the prediction horizon. When the prediction horizon is large, other effective methods need to be considered.

In addition, we can also directly consider the ball boundary problem. For the defined neighborhood ball set $B(\mathbf{0}; r)$, the support function with respect to the set is given by:

$$h_B(x) = \max_{\mathbf{w}_k \in B(\mathbf{0}; r)} x^T \mathbf{w}_k \tag{32}$$

Meanwhile, we can simplify (29) as:

$$\Gamma^T \mathbf{w}_k + \Upsilon \leq 0 \tag{33}$$

where Γ and Υ are both affine functions of the optimization variables. Comparing these two equations, we obtain the following inequality constraint:

$$h_B(\Gamma) \leq -\Upsilon \tag{34}$$

It is easy to prove that for the ball set $B(\mathbf{0}; r)$, the value of the function $h_B(\Gamma)$ is $r \|\Gamma\|_2$. Therefore, the original MPC problem can be transformed into a conic program optimization problem with second-order cone constraints.

IV. Results

In this section, we first introduce the nominal model and compare it with the model in [17]. Then, we compare the distribution generated by the proposed flow-based model to the real uncertainty samples. Finally, we apply a standard and stochastic MPC to the GCAI control problem.

A. *Nominal Model*

A simple three-layer fully connected neural network with skip connections in each layer is used as a nominal model. The model inputs (NVO, fuel, and water injection durations) and outputs (CA50, IMEP, and MPRR) have been described in Section II. The dataset consists of $53,000$ combustion cycles, divided into training, validation, and test datasets with $40,000$, $10,000$, and 3000 samples, respectively. All samples are shuffled to ensure sample independence. The performance results are compared to those of an experimental model in [17] with an LSTM structure.

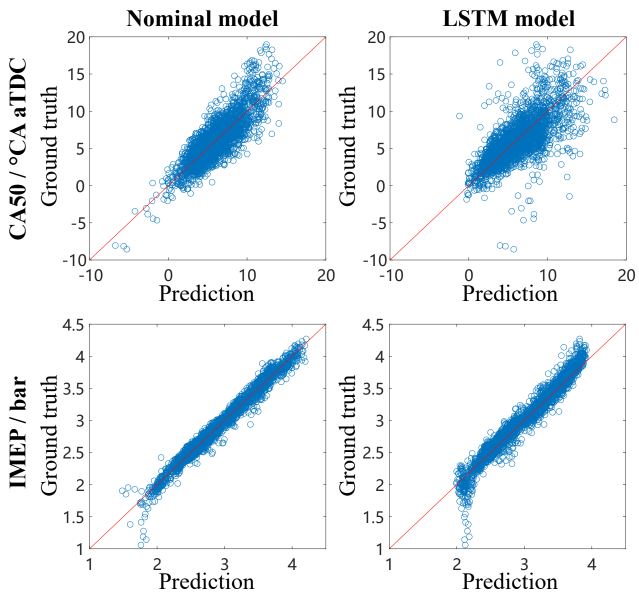

Fig. 3. Qualitative comparison of the nominal and baseline models in [17]. The prediction results of CA50 and IMEP by two methods are depicted in the first and second rows. The horizontal axis represents the predicted values in each subfigure, while the vertical axis represents the ground truth.

The comparison of the nominal model and the baseline model is shown in Fig. 3. As we focus on the fitting capacity of CA50 and IMEP, Fig. 3 shows the prediction of these two engine parameters. The closer the sample points are to the red line, the better the model fits the variable. We can see the points of the nominal model are clustered closer to the red line, showing a better prediction on both CA50 and IMEP compared to the LSTM model.

In Table I, the coefficient of determination (R^2) and normalized root mean squared error (NRMSE) with the interquartile range as normalization are displayed.

TABLE I
QUANTITATIVE COMPARISON OF ENGINE PARAMETERS.

	nominal model	baseline model (LSTM)
CA50 (R^2)	0.755	0.528
IMEP (R^2)	0.971	0.945
CA50 (NRMSE)	0.484	0.673
IMEP (NRMSE)	0.105	0.144

Taking a closer look at the quantitative results, we can conclude that the nominal model shows a better fitting ability for both CA50 and IMEP compared to the LSTM model. It is notable that the nominal model significantly improves the prediction performance of CA50. Although both models perform well in predicting IMEP in quantitative analysis, we can see in Fig. 3 that the LSTM model has a clear barrier in predicting low IMEP values.

B. *Generative model*

The previous subsection demonstrates that the nominal model can fit complex systems well. Based on the previous findings, the residuals between the predicted data of the nominal model and the ground truth are used as training and validation datasets for the flow-based generative model. In addition, $10,000$ data pairs, including the states and residuals, are used for testing. As introduced before, the generative model is fed with samples from a standard normal distribution and the current states to generate the approximated residual samples.

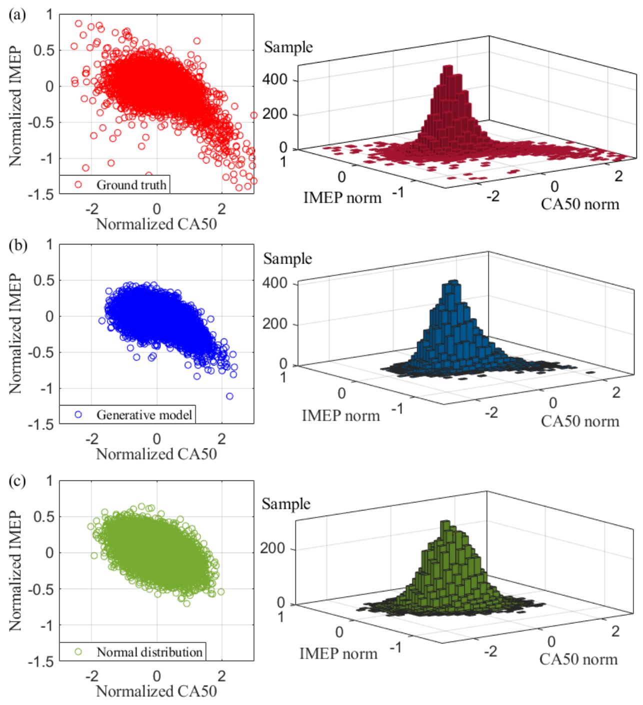

Fig. 4. Comparison among (a) the real uncertainty distribution, (b) the generated distribution, and (c) the fitted normal distribution.

The generated data and the real residuals are shown in Fig 4. To provide a detailed comparison of different distributions, we additionally use a fitted normal distribution. The normalization parameters for CA50 and IMEP are 2.96 °CA and 0.51 bar, respectively.

As shown in Fig 4, the real residual uncertainty shows a non-standard distribution, which is asymmetric about the mean, meaning that the skewness is not equal to zero. It can also be seen that the kurtosis of the residuals is higher than that of a normal distribution, indicating a sharper peak and a steeper shape. Estimating the first and second moments of the distribution is not sufficient to accurately describe the details of the real uncertainty distribution. For example, there are more outliers in the bottom right of the real distribution, but the probability density compared to the center is relatively low. These points attract more concerns as they may be caused by combustion irregularities, which are hard to predict. Thus, it is impractical to use a simple normal distribution to reflect the complex fact.

However, the generative model fits the real uncertainty distribution well. The distribution shape of the generated samples is similar to the ground truth. Therefore, we can quickly obtain samples of the target distributions through the generative model. Nevertheless, the training of the model is sensitive to hyperparameters. Thus, we expect a more proper fitting of distribution by further fine-tuning. Currently, this generative model only considers two variables: CA50 and IMEP. Therefore, uncertainties are added to these two variables in the later subsection.

C. Stochastic MPC

So far, we have performed learning-based modeling of the complete system, including the nominal model and the generative model for uncertainty prediction. In this section, we consider both of them in sMPC strategies.

We first linearize the models for each cycle. It is easy to calculate the Jacobian matrices based on the backpropagation rule of the neural network. We choose a prediction horizon of $N = 3$, the confidence probability of 90% and 95% for each state and input constraint, and the cost weights as identity matrices concerning all the normalized variables. For comparison, we use a standard MPC, which is also applied to the linearized models and a setpoint of 7 °CA aTDC for CA50 to ensure a stable combustion efficiency. The reference of IMEP is variable, which is related to the torque requirement. We perform 50 simulations for both MPCs, and the results are shown in Fig 5 and Fig 6.

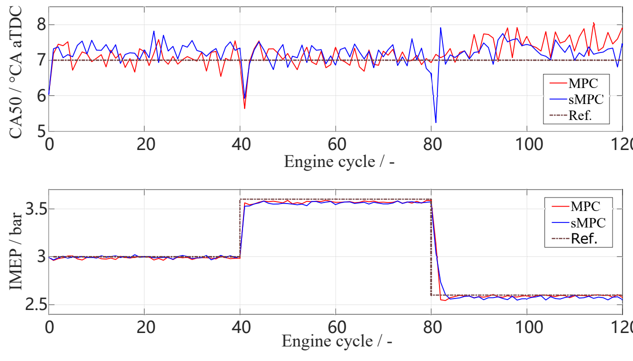

Fig. 5. Performance comparison of CA50 and IMEP with sMPC and the standard MPC. The blue lines and red lines represent the average value of 50 simulations for sMPC and MPC, respectively. The reference is marked as brown lines.

Fig 5 depicts the average control performance of CA50 and IMEP in 50 simulations with the sMPC and the standard MPC. In the first two-thirds of the cycles, the MPC performs slightly better than sMPC concerning the mean values with lower control deviations, where the engine operates at proper load conditions. At the same time, MPC has better transient performance, with no significant variation in CA50. However, in the remaining cycles, the proposed sMPC outperforms MPC obviously in CA50 due to the influence of the uncertainties at the boundary load condition. We can see more details in Fig 6, which reports the variation of CA50 in 50 simulations with sMPC and MPC. The variation range with the MPC is larger than the proposed sMPC, especially in the remaining one-third of the cycles. As delayed combustion could lead to irregular outliers in consecutive cycles, large variations are undesirable. Similar results are reflected in the quantitative statistics of CA50 in Table II.

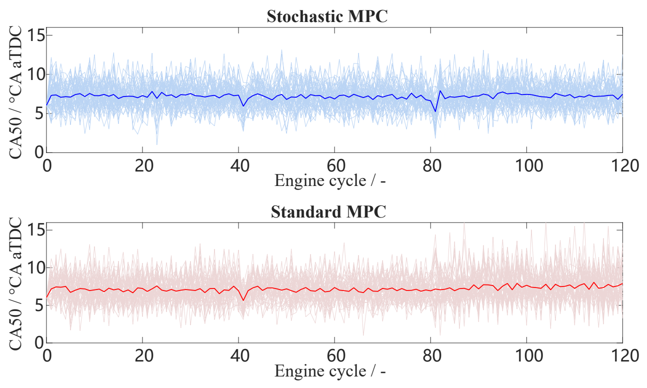

Fig. 6. Qualitative variation comparison of CA50 with the sMPC and the standard MPC in 50 simulations. The blue and red regions show the various samples of the sMPC and MPC, respectively. The dark-colored lines present the mean value of the samples.

TABLE II
QUANTITATIVE COMPARISON OF CA50 WITH THE SMPC AND MPC.

	variance	ratio (>11 °CA)	ratio (<3 °CA)
sMPC	2.74	1.79%	0.38%
MPC	3.59	3.37%	0.73%

V. CONCLUSION

This work proposes a generative model to learn the uncertainty distribution between the model and experimental data. Based on this model, we implement a less restrictive stochastic MPC strategy with affine disturbance feedback. Meanwhile, we transform the original optimization problem with chance constraints into tractable optimization problems. The proposed generative model is able to achieve the performance that simple distribution approximation cannot, such as fitting the skewness and kurtosis of the distribution. Furthermore, the generative model provides prior knowledge about uncertainties for controllers, improving the control performance with smaller output variances. However, the current work still has some limitations. We do not make full use of the generative model's capabilities as the current assumption about the conditional distribution of uncertainties given the states is a normal distribution due to the linearization. Next, we will consider ways to utilize the nonlinear information implied by the generative model. For instance, we can use random samples to solve scenario optimization problem.

VI. Appendix

The Matrices H_x, H_u and H_w are given as:

$$H_x = \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, H_u = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix},$$
$$H_w = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ D & 0 & \cdots & 0 \\ AD & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}D & A^{N-2}D & \cdots & D \end{bmatrix} \tag{35}$$

Then, we consider the following inequality:

$$F_{xj}x_k + F_{uj}\mathbf{h}_k + F_{uj}\mathbf{V}_k\mathbf{w}_k + F_{wj}\mathbf{w}_k - \mathbf{f}_j \leq 0 \tag{36}$$

where $F_{xj} = \mathbf{F}_j H_x$, $F_{uj} = \mathbf{F}_j H_u$, $F_{wj} = \mathbf{F}_j H_w$, $\mathbf{h}_k$ and $\mathbf{V}_k$ are optimization variables, and $\mathbf{w}_k$ is the disturbance sequence. Next, we consider each $V_{q,p|k}$ in $\mathbf{V}_k$ and rewrite $F_{uj}\mathbf{V}_k$ as:

$$F_{uj}\mathbf{V}_k = \sum_{p=1}^{N-1}\sum_{q=0}^{p-1} F_{uj}\left(E_{p,q} \otimes V_{q,p|k}\right) \tag{37}$$

with $F_{uj} \in \mathbb{R}^{1\times n_u N}$, $V_{q,p|k} \in \mathbb{R}^{n_u \times n_w}$, and $E_{p,q} \in \mathbb{R}^{N\times N}$. The matrix $E_{p,q}$ has a value of one at $(p+1, q+1)$ and zeros elsewhere. Then, we can make the following transformation:

$$\begin{aligned} \left(F_{uj}\left(E_{p,q} \otimes V_{q,p|k}\right)\right)^T &= \left(E_{p,q}^T \otimes V_{q,p|k}^T\right) F_{uj}^T \\ = vec\left(V_{q,p|k}^T \tilde{F} E_{p,q}\right) &= vec(I_{n_w\times n_w} V_{q,p|k}^T \tilde{T}) \\ = (\tilde{T}^T \otimes I_{n_w\times n_w}) vec(V_{q,p|k}^T) &= T_{p,q} vec(V_{q,p|k}^T) \end{aligned} \tag{38}$$

where $vec(\cdot)$ denotes the vectorization of the matrix into a column vector, $vec(\tilde{F}) = F_{uj}^T$ with $\tilde{F} \in \mathbb{R}^{n_u\times N}$, $\tilde{T} = \tilde{F}E_{p,q}$ with $\tilde{T} \in \mathbb{R}^{n_u\times N}$, and $T_{p,q} = \tilde{T}^T \otimes I_{n_w\times n_w}$ with $T_{p,q} \in \mathbb{R}^{n_w N\times n_w n_u}$. Then, (37) can be written as:

$$\begin{gathered} \mathbf{V}_k^T F_{uj}^T = [T_{1,0}, T_{2,1}, ..., T_{N-1,N-2}] \\ \left[row(V_{0,1|k}), row(V_{1,2|k}), ..., row(V_{N-2,N-1|k})\right]^T \\ = T_j \mathbf{s}_k \end{gathered} \tag{39}$$

where $row(\cdot)$ is the row expansion function of a matrix, $T_j \in \mathbb{R}^{n_w N\times n_w n_u \frac{N(N-1)}{2}}$, and $\mathbf{s}_k \in \mathbb{R}^{n_w n_u \frac{N(N-1)}{2}}$ is the extended vector form of $\mathbf{V}_k$.

VII. Acknowledgments

The presented work was performed as part of the Research Unit (Forschungsgruppe) FOR 2401 "Optimization-based Multiscale Control for Low Temperature Combustion Engines", which is funded by the German Research Association (Deutsche Forschungsgemeinschaft, DFG). The support is gratefully acknowledged.

References

[1] B. Lehrheuer, B. Morcinkowski, S. Pischinger, and M. Nijs, "Low temperature gasoline combustion – potential, challenges, process modeling and control," in *Active Flow and Combustion Control 2014*, R. King, Ed. Cham: Springer International Publishing, 2015, pp. 163–179.

[2] M. Sjöberg, J. E. Dec, and W. Hwang, "Thermodynamic and chemical effects of egr and its constituents on hcci autoignition," *SAE Transactions*, vol. 116, pp. 271–289, 2007.

[3] M. Wick, B. Lehrheuer, T. Albin, J. Andert, and S. Pischinger, "Decoupling of consecutive gasoline controlled auto-ignition combustion cycles by field programmable gate array based real-time cylinder pressure analysis," *International Journal of Engine Research*, vol. 19, no. 2, pp. 153–167, 2017.

[4] E. Hellström, A. Stefanopoulou, J. Vavra, A. Babajimopoulos, D. Assanis, L. Jiang, and H. Yilmaz, "Understanding the dynamic evolution of cyclic variability at the operating limits of hcci engines with negative valve overlap," *SAE International Journal of Engines*, vol. 5, no. 3, pp. 995–1008, 2012.

[5] M. Fathi, O. Jahanian, and M. Shahbakhti, "Modeling and controller design architecture for cycle-by-cycle combustion control of homogeneous charge compression ignition (hcci) engines – a comprehensive review," *Energy Conversion and Management*, vol. 139, pp. 1–19, 2017.

[6] Javad Rezaei, Mahdi Shahbakhti, Bahram Bahri, and Azhar Abdul Aziz, "Performance prediction of hcci engines with oxygenated fuels using artificial neural networks," *Applied Energy*, vol. 138, pp. 460–473, 2015.

[7] H. Bendu, B. Deepak, and S. Murugan, "Application of grnn for the prediction of performance and exhaust emissions in hcci engine using ethanol," *Energy Conversion and Management*, vol. 122, pp. 165–173, 2016.

[8] M. Taghavi, A. Gharehghani, F. B. Nejad, and M. Mirsalim, "Developing a model to predict the start of combustion in hcci engine using ann-ga approach," *Energy Conversion and Management*, vol. 195, pp. 57–69, 2019.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[11] L. Dinh, D. Krueger, and Y. Bengio, "Nice: Non-linear independent components estimation," *arXiv preprint arXiv:1410.8516*, 2014.

[12] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," *Advances in neural information processing systems*, vol. 31, 2018.

[13] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523–533, 2006.

[14] J. Lofberg, "Approximations of closed-loop minimax mpc," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 2. IEEE, 2003, pp. 1438–1442.

[15] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, "Adjustable robust solutions of uncertain linear programs," *Mathematical programming*, vol. 99, no. 2, pp. 351–376, 2004.

[16] D. H. van Hessem and O. H. Bosgra, "A conic reformulation of model predictive control including bounded and stochastic disturbances under state and input constraints," in *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, vol. 4. IEEE, 2002, pp. 4643–4648.

[17] A. Norouzi, S. Shahpouri, D. Gordon, A. Winkler, E. Nuss, D. Abel, J. Andert, M. Shahbakhti, and C. Koch, "Deep learning based model predictive control for compression ignition engines," *Control Engineering Practice*, vol. 127, p. 105299, 2022.

[18] L. Koopmans, R. Ogink, and I. Denbratt, "Direct gasoline injection in the negative valve overlap of a homogeneous charge compression ignition engine," *SAE Transactions*, vol. 112, pp. 1365–1376, 2003.